Circuit Evaluation for Finite Semirings

Moses Ganardi, Danny Hucke, Daniel König, Markus Lohrey

University of Siegen, Germany
{ganardi,hucke,koenig,lohrey}@eti.uni-siegen.de

Abstract

The computational complexity of the circuit evaluation problem for finite semirings is considered, where semirings are not assumed to have an additive or multiplicative identity. The following dichotomy is shown: If a finite semiring is such that (i) the multiplicative semigroup is solvable and (ii) it does not contain a subsemiring with an additive identity 0 and a multiplicative identity $1 \neq 0$, then the circuit evaluation problem for the semiring is in $\mathsf{DET} \subseteq \mathsf{NC}^2$. In all other cases, the circuit evaluation problem is P-complete.

1 Introduction

Circuit evaluation problems are among the most well-studied computational problems in complexity theory. In its most general formulation, one has an algebraic structure $\mathcal{A} = (A, f_1, \ldots, f_k)$, where the f_i are mappings $f_i : A^{n_i} \to A$. A circuit over the structure $\mathcal{A}$ is a directed acyclic graph (dag) where every inner node is labelled with one of the operations f_i and has exactly n_i incoming edges that are linearly ordered. The leaf nodes of the dag are labelled with elements of A (for this, one needs a suitable finite representation of elements from A), and there is a distinguished output node. The task is to evaluate this dag in the natural way, and to return the value of the output node.

In his seminal paper [20], Ladner proved that the circuit evaluation problem for the Boolean semiring $\mathbb{B}_2 = (\{0,1\}, \vee, \wedge)$ is P-complete. This result marks a cornerstone in the theory of P-completeness [14], and motivated the investigation of circuit evaluation problems for other algebraic structures. A large part of the literature is focused on arithmetic (semi)rings like $(\mathbb{Z}, +, \cdot)$, $(\mathbb{N}, +, \cdot)$ or the max-plus semiring $(\mathbb{Z} \cup \{-\infty\}, \max, +)$ [2, 3, 19, 25, 26, 35]. These papers mainly consider circuits of polynomial formal degree. For commutative semirings, circuits of polynomial formal degree can be restructured into an equivalent (unbounded fan-in) circuit of polynomial size and logarithmic depth [35]. This result leads to NC-algorithms for evaluating polynomial degree circuits over commutative semirings [25, 26]. Over non-commutative semirings, circuits of polynomial formal degree do in general not allow a restructuring into circuits of logarithmic depth [19].

In [26] it was shown that also for finite non-commutative semirings circuit evaluation is in NC for circuits of polynomial formal degree. On the other hand, the authors are not aware of any NC-algorithms for evaluating general (exponential degree) circuits over semirings. The lack of such algorithms is probably due to Ladner's result, which seems to exclude any efficient parallel algorithms. On the other hand, in the context of semigroups, there exist NC-algorithms for circuit evaluation. In [10], the following dichotomy result was shown for finite semigroups: If the finite semigroup is solvable (meaning that every subgroup is a solvable group), then circuit evaluation is in NC (in fact, in DET, which is the class of all problems that are AC^0-reducible to the computation of an integer determinant [11, 12]), otherwise circuit evaluation is P-complete.

In this paper, we extend the work of [10] from finite semigroups to finite semirings. On first sight, it seems again that Ladner's result excludes efficient parallel algorithms: It is not hard to show that if the finite semiring has an additive identity 0 and a multiplicative identity $1 \neq 0$

(where 0 is not necessarily absorbing with respect to multiplication), then circuit evaluation is P-complete, see Lemma 4.5. Therefore, we take the most general reasonable definition of semirings: A semiring is a structure $(R, +, \cdot)$, where $(R, +)$ is a commutative semigroup, $(R, \cdot)$ is a semigroup, and $\cdot$ distributes (on the left and right) over $+$. In particular, we neither require the existence of a 0 nor a 1. Our main result states that in this general setting there are only two obstacles to efficient parallel circuit evaluation: non-solvability of the multiplicative structure and the existence of a zero and a one (different from the zero) in a subsemiring. More precisely, we show the following two results, where a semiring is called $\{0, 1\}$-free if there exists no subsemiring in which an additive identity 0 and a multiplicative identity $1 \neq 0$ exist:

(1) If a finite semiring is not $\{0, 1\}$-free, then the circuit evaluation problem is P-complete.

(2) If a finite semiring $(R, +, \cdot)$ is $\{0, 1\}$-free, then the circuit evaluation problem for $(R, +, \cdot)$ can be solved with AC^0-circuits that are equipped with oracle gates for (a) graph reachability, (b) the circuit evaluation problem for the commutative semigroup $(R, +)$ and (c) the circuit evaluation problem for the semigroup $(R, \cdot)$.

Together with the dichotomy result from [10] (and the fact that commutative semigroups are solvable) we get the following result: For every finite semiring $(R, +, \cdot)$, the circuit evaluation problem is in NC (in fact, in DET) if $(R, \cdot)$ is solvable and $(R, +, \cdot)$ is $\{0, 1\}$-free. Moreover, if one of these conditions fails, then circuit evaluation is P-complete.

The hard part of the proof is to show the above statement (2). We will proceed in two steps. In the first step we reduce the circuit evaluation problem for a finite semiring R to the evaluation of a so called type admitting circuit. This is a circuit where every gate evaluates to an element of the form eaf, where e and f are multiplicative idempotents of R. Moreover, these idempotents e and f have to satisfy a certain compatibility condition that will be expressed by a so called type function. In a second step, we present a parallel evaluation algorithm for type admitting circuits. Only for this second step we need the assumption that the semiring is $\{0, 1\}$-free.

In Section 6 we present an application of our main result for circuit evaluation to formal language theory. We consider the intersection non-emptiness problem for a given context-free language and a fixed regular language L. If the context-free language is given by an arbitrary context-free grammar, then we show that the intersection non-emptiness problem is P-complete as long as L is not empty (Theorem 6.1). It turns out that the reason for this is non-productivity of nonterminals. We therefore consider a restricted version of the intersection non-emptiness problem, where every nonterminal of the input context-free grammar must be productive. To avoid a promise problem (testing productivity of a nonterminal is P-complete), we in addition provide a witness of productivity for every nonterminal. This witness consists of exactly one production $A \to w$ for every nonterminal of A such that the set of all the set of all selected productions is an acyclic grammar $\mathcal{H}$. This ensures that $\mathcal{H}$ derives for every nonterminal A exactly one string that is a witness of the productivity of A. We then show that this restricted version of the intersection non-emptiness problem with the fixed regular language L is equivalent (with respect to constant depth reductions) to the circuit evaluation problem for a certain finite semiring that is derived from the syntactic monoid of the regular language L.

Further related work We mentioned already the existing work on circuit evaluation for infinite semirings. The question whether a given circuit over a polynomial ring evaluates to the zero polynomial is also known as polynomial identity testing. For polynomial rings over $\mathbb{Z}$ or $\mathbb{Z}_n$ ($n \geq 2$), polynomial identity testing has a co-randomized polynomial time algorithm [1, 15]. Moreover, the question, whether a deterministic polynomial time algorithm exists is tightly related to lower bounds in complexity theory, see [31] for a survey. For infinite groups, the circuit evaluation problem is also known as the compressed word problem [21]. In the context of parallel algorithms, it is interesting to note that the third and forth author recently proved that the circuit evaluation problem for finitely generated (but infinite) nilpotent groups belongs to DET [17]. For finite non-associative groupoids, the complexity of circuit evaluation was studied in [27], and some of the results from [10] for semigroups were generalized to the non-associative setting. In [8], the problem

of evaluating tensor circuits is studied. The complexity of this problem is quite high: Whether a given tensor circuit over the Boolean semiring evaluates to the (1×1)-matrix (0) is complete for nondeterministic exponential time. Finally, let us mention the papers [24, 34], where circuit evaluation problems are studied for the power set structures $(2^{\mathbb{N}}, +, \cdot, \cup, \cap, ^-)$ and $(2^{\mathbb{Z}}, +, \cdot, \cup, \cap, ^-)$, where $+$ and $\cdot$ are evaluated on sets via $A \circ B = \{a \circ b \mid a \in A, b \in B\}$.

A variant of our intersection non-emptiness problem was studied in [30]. There, a context-free language L is fixed, a (deterministic or non-deterministic) finite automaton $\mathcal{A}$ is the input, and the question is, whether $L \cap L(\mathcal{A}) = \emptyset$ holds. The authors present large classes of context-free languages such that for each member the intersection non-emptiness problem with a given regular language (specified by a non-deterministic automaton) is P-complete (resp., NL-complete).

2 Computational complexity

For background in complexity theory the reader might consult [6]. We assume that the reader is familiar with the complexity classes NL (non-deterministic logspace) and P (deterministic polynomial time). A function is logspace-computable if it can be computed by a deterministic Turing-machine with a logspace-bounded work tape, a read-only input tape, and a write-only output tape. Note that the logarithmic space bound only applies to the work tape. P-hardness will refer to logspace reductions.

We use standard definitions concerning circuit complexity, see e.g. [37]. We only consider polynomially bounded families $(C_n)_{n\geq 0}$ of Boolean circuits, where the number of gates of C_n is bounded by a polynomial $p(n)$. For such a family, gates of C_n can be encoded with bit strings of length $O(\log n)$. The family $(C_n)_{n\geq 0}$ is $\mathsf{DLOGTIME}$-uniform, if for given binary coded gates u, v of C_n, one can (i) compute the type of gate u in time $O(\log n)$ and (ii) check in time $O(\log n)$ whether u is an input gate for v. Note that the time bound $O(\log n)$ is linear in the input length $|u|+|v|$. All circuit families in this paper are implicitly assumed to be $\mathsf{DLOGTIME}$-uniform. We will consider the class AC^0 of all problems that can be recognized by a polynomial size circuit family of constant depth built up from NOT-gates (which have fan-in one) and AND- and OR-gates of unbounded fan-in. The class NC^k $(k \geq 1)$ is defined by polynomial size circuit families of depth $O(\log^k n)$ that use NOT-gates, and AND- and OR-gates of fan-in two. One defines $\mathsf{NC} = \bigcup_{k\geq 1} \mathsf{NC}^k$. The above language classes can be easily generalized to classes of functions by allowing circuits with several output gates. Of course, this only allows to compute functions $f : \{0,1\}^* \to \{0,1\}^*$ such that $|f(x)| = |f(y)|$ whenever $|x| = |y|$. If this condition is not satisfied, one has to consider a suitably padded version of f.

We use the standard notion of constant depth Turing-reducibility: For functions $f_1, \ldots, f_k$ let $\mathsf{AC}^0(f_1, \ldots, f_k)$ be the class of all functions that can be computed with a polynomial size circuit family of constant depth that uses NOT-gates and unbounded fan-in AND-gates, OR-gates, and f_i-oracle gates $(1 \leq i \leq k)$. Here, an f_i-oracle gate receives an ordered tuple of inputs $x_1, x_2, \ldots, x_n$ and outputs the bits of $f_i(x_1 x_2 \cdots x_n)$. By taking the characteristic function of a language, we can also allow a language $L_i \subseteq \{0,1\}^*$ in place of f_i. Note that the function class $\mathsf{AC}^0(f_1, \ldots, f_k)$ is closed under composition (since the composition of two AC^0-circuits is again an AC^0-circuit). We write $\mathsf{AC}^0(\mathsf{NL}, f_1, \ldots, f_k)$ for $\mathsf{AC}^0(\mathsf{GAP}, f_1, \ldots, f_k)$, where GAP is the NL-complete graph accessibility problem. The class $\mathsf{AC}^0(\mathsf{NL})$ is studied in [5]. It has several alternative characterizations and can be viewed as a nondeterministic version of functional logspace. As remarked in [5], the restriction of $\mathsf{AC}^0(\mathsf{NL})$ to 0-1 functions is NL. Clearly, every logspace-computable function belongs to $\mathsf{AC}^0(\mathsf{NL})$: The NL-oracle can be used to directly compute the output bits of a logspace-computable function.

Let $\mathsf{DET} = \mathsf{AC}^0(\mathsf{det})$, where det is the function that maps a binary encoded integer matrix to the binary encoding of its determinant, see [11]. Actually, Cook defined DET as $\mathsf{NC}^1(\mathsf{det})$ [11], but the above definition via AC^0-circuits seems to be more natural. For instance, it implies that DET is equal to the #L-hierarchy, see also the discussion in [12].

We defined DET as a function class, but the definition can be extended to languages by considering their characteristic functions. It is well known that $\mathsf{NL} \subseteq \mathsf{DET} \subseteq \mathsf{NC}^2$ [12]. From $\mathsf{NL} \subseteq \mathsf{DET}$,

it follows easily that $\mathsf{AC}^0(\mathsf{NL}, f_1, \ldots, f_k) \subseteq \mathsf{DET}$ whenever $f_1, \ldots, f_k \in \mathsf{DET}$.

3 Algebraic structures, semigroups, and semirings

An *algebraic structure* $\mathcal{A} = (A, f_1, \ldots, f_k)$ consists of a non-empty *domain* A and operations $f_i : A^{n_i} \to A$ for $1 \leq i \leq k$. We often identify the domain with the structure, if it is clear from the context. A *substructure* of $\mathcal{A}$ is a subset $B \subseteq A$ that closed under each of the operations f_i. We identify B with the structure $(B, g_1, \ldots, g_k)$, where $g_i : B^{n_i} \to B$ is the restriction of f_i to B^{n_i} for all $1 \leq i \leq k$.

We mainly deal with semigroups and semirings. In the following two subsection we present the necessary background. For further details on semigroup theory (resp., semiring theory) see [29] (resp., [13]).

3.1 Semigroups

A *semigroup* $(S, \circ)$(or just S) is an algebraic structure with a single associative binary operation. We usually write st for $s \circ t$. If $st = ts$ for all $s, t \in S$, we call S *commutative*. A set $I \subseteq S$ is called a *semigroup ideal* if for all $s \in S$, $a \in I$ we have $sa, as \in I$. An element $e \in S$ is called *idempotent* if $ee = e$. It is well-known that for every finite semigroup S and $s \in S$ there exists an $n \geq 1$ such that s^n is idempotent. In particular, every finite semigroup contains an idempotent element. By taking the smallest common multiple of all these n, one obtains an $\omega \geq 1$ such that s^ω is idempotent for all $s \in S$. The set of all idempotents of S is denoted with $E(S)$. If S is finite, then $SE(S)S = S^n$ where $n = |S|$. Moreover, $S^n = S^m$ for all $m \geq n$. For a set Σ, the *free semigroup* generated by Σ is the set Σ^+ of all finite non-empty words over Σ together with the operation of concatenation.

A semigroup M with an identity element $1 \in M$, i.e. $1m = m1 = m$ for all $m \in M$, is called a *monoid*. With S^1 we denote the monoid that is obtained from a semigroup S by adding a fresh element 1, which becomes the identity element of S^1. Thus, we extend the multiplication to $S^1 = S \cup \{1\}$ by setting $1s = s1 = s$ for all $s \in S \cup \{1\}$. In case M is monoid and N is a submonoid of M, we do not require that the identity element of N is the identity element of M. But, clearly, the identity element of the submonoid N must be an idempotent element of M. In fact, for every semigroup S and every idempotent $e \in E(S)$, the set $eSe = \{ese \mid s \in S\}$ is a submonoid of S with identity e, which is also called a *local submonoid* of S. The local submonoid eSe is the maximal submonoid of S whose identity element is e. A semigroup S is *aperiodic* if every subgroup of S is trivial. A semigroup S is *solvable* if every subgroup G of S is a solvable group, i.e., for the series defined by $G_0 = G$ and $G_{i+1} = [G_i, G_i]$ (the commutator subgroup of G_i) there exists an $i \geq 0$ with $G_i = 1$. Since Abelian groups are solvable, every commutative semigroup is solvable.

3.2 Semirings

A *semiring* $(R, +, \cdot)$ consists of a non-empty set R with two operations $+$ and $\cdot$ such that $(R, +)$ is a commutative semigroup, $(R, \cdot)$ is a semigroup, and $\cdot$ left- and right-distributes over $+$, i.e., $a \cdot (b + c) = ab + ac$ and $(b + c) \cdot a = ba + ca$ (as usual, we write ab for $a \cdot b$). Note that we neither require the existence of an additive identity 0 nor the existence of a multiplicative identity 1. We denote with $R_+ = (R, +)$ the additive semigroup of R and with $R_\bullet = (R, \cdot)$ the multiplicative semigroup of R. For $n \geq 1$ and $r \in R$ we write $n \cdot r$ or just nr for $r + \cdots + r$, where r is added n times. With $E(R)$ we denote the set of multiplicative idempotents of R, i.e., those $e \in R$ with $e^2 = e$. Note that for every multiplicative idempotent $e \in E(R)$, eRe is a subsemiring of R in which the multiplicative structure is a monoid. For a non-empty subset $T \subseteq R$ we denote by $\langle T \rangle$ the subsemiring generated by T, i.e. the smallest set containing T which is closed under addition and multiplication. An *ideal* of R is a subset $I \subseteq R$ such that for all $a, b \in I$, $s \in R$ we have $a + b, sa, as \in I$. Clearly, every ideal is a subsemiring. Let $\mathbb{B}_2 = (\{0, 1\}, \vee, \wedge)$ be the *Boolean semiring*.

For a given non-empty set Σ, the *free semiring* $\mathbb{N}[\Sigma]$ generated by Σ consists of all mappings $f : \Sigma^+ \to \mathbb{N}$ such that $\mathsf{supp}(f) := \{w \in \Sigma^+ \mid f(w) \neq 0\}$ is finite and non-empty. Addition is defined pointwise, i.e., $(f+g)(w) = f(w)+g(w)$, and multiplication is defined by the convolution: $(f \cdot g)(w) = \sum_{w=uv} f(u) \cdot g(v)$, where the sum is taken over all factorizations $w = uv$ with $u, v \in \Sigma^+$. We view an element $f \in \mathbb{N}[\Sigma]$ as a non-commutative polynomial $\sum_{w \in \mathsf{supp}(f)} f(w) \cdot w$. Then addition (resp. multiplication) in $\mathbb{N}[\Sigma]$ corresponds to addition (resp. multiplication) of non-commutative polynomials. Words $w \in \mathsf{supp}(f)$ are also called *monomials* of f. A word $w \in \Sigma^+$ is identified with the non-commutative polynomial $1 \cdot w$, i.e., the mapping f with $\mathsf{supp}(f) = \{w\}$ and $f(w) = 1$. For every semiring R which is generated by Σ there exists a canonical surjective homomorphism from $\mathbb{N}[\Sigma]$ to R which evaluates non-commutative polynomials over Σ. Since a semiring is not assumed to have a multiplicative identity (resp., additive identity), we have to exclude the empty word from $\mathsf{supp}(f)$ for every $f \in \mathbb{N}[\Sigma]$ (resp., exclude the mapping f with $\mathsf{supp}(f) = \emptyset$ from $\mathbb{N}[\Sigma]$).

A crucial definition in this paper is that of a $\{0,1\}$-*free* semiring. This is a semiring R which does *not* contain a subsemiring T with an additive identity 0 and a multiplicative identity $1 \neq 0$. Note that in such a semiring T we do not require that 0 is absorbing, i.e., $a \cdot 0 = 0 \cdot a = 0$ for all $a \in T$. The class of $\{0,1\}$-free *finite* semirings has several characterizations:

Lemma 3.1. *For a finite semiring R, the following are equivalent:*

1. *R is not $\{0,1\}$-free.*
2. *R contains $\mathbb{B}_2$ or the ring $\mathbb{Z}_d$ for some $d \geq 2$ as a subsemiring.*
3. *R is divided by $\mathbb{B}_2$ or $\mathbb{Z}_d$ for some $d \geq 2$ (i.e., $\mathbb{B}_2$ or $\mathbb{Z}_d$ is a homomorphic image of a subsemiring of R).*
4. *There exist elements $0, 1 \in R$ such that $0 \neq 1$, $0+0=0$, $0+1=1$, $0 \cdot 1 = 1 \cdot 0 = 0 \cdot 0 = 0$, and $1 \cdot 1 = 1$ (but $1+1 \neq 1$ is possible).*

Proof. $(1 \Rightarrow 2)$: Let T be a subsemiring of R which has a zero element 0 and a one element $1 \neq 0$. Note that $0 \cdot 0 = 0 \cdot 0 + 0 = 0 \cdot 0 + 1 \cdot 0 = (0+1) \cdot 0 = 1 \cdot 0 = 0$. Let $T' = \{0\} \cup \{k \cdot 1 \mid k \in \mathbb{N}\}$, which is the subsemiring generated by these elements. It is isomorphic to some semiring $B(k,d)$ $(k \geq 0,\ d \geq 1)$, which is the semiring $(\mathbb{N}, +, \cdot)$ modulo the congruence relation $\sim$ defined by $i \sim j$ if $0 \leq i = j \leq k-1$ or $(i, j \geq k$ and d divides $i-j)$. Since $0 \neq 1$, we have $(k,d) \neq (0,1)$. If $k = 0$, then $B(0,d)$ is isomorphic to $\mathbb{Z}_d$ for $d \geq 2$. If $k \geq 1$, then choose $a \geq k$ such that d divides a, for example $a = dk$. Then $\{0, a \cdot 1\}$ is a subsemiring isomorphic to the Boolean semiring $\mathbb{B}_2$.

$(2 \Rightarrow 3)$: This implication is trivial.

$(3 \Rightarrow 4)$: Assume that $\varphi : T \to T'$ is a homomorphism from a subsemiring T of R to T', where the latter is $\mathbb{B}_2$ or $\mathbb{Z}_d$ with $d \geq 2$. In particular, there exist $0, 1 \in T'$ with $0 \neq 1$, $0+0=0$, $0+1=1$, $0 \cdot 1 = 1 \cdot 0 = 0 \cdot 0 = 0$, and $1 \cdot 1 = 1$. Let $n \geq 1$ be such that $n \cdot x$ is additively idempotent and x^n is multiplicatively idempotent for all $x \in R$. Then $n \cdot x^n$ is is additively and multiplicatively idempotent for all $x \in R$. Let $a, e \in T$ be such that $\varphi(a) = 0$ and $\varphi(e) = 1$. We can replace a by $n \cdot a^n$ and e by e^n. Then, $a + a = aa = a$ and $ee = e$. For $a' = n \cdot (eae)^n$ we have $\varphi(a') = 0$ and $a'e = ea' = a' + a' = a'a' = a'$. For $e' = a' + e$ we have $\varphi(e') = 1$ (hence, $a' \neq e'$) and $e'e' = a'a' + a'e + ea' + ee = a' + e = e'$, $a' + e' = a' + a' + e = e'$. Furthermore, we have $a'e' = a'(a' + e) = a' + a'e = a'$ and similarly $e'a' = a'$. Hence, a' and e' satisfy all equations from point 4.

$(4 \Rightarrow 1)$: Assume that there exist elements $0, 1 \in R$ such that $0 \neq 1$, $0+0=0$, $0+1=1$, $0 \cdot 1 = 1 \cdot 0 = 0 \cdot 0 = 0$, and $1 \cdot 1 = 1$. Consider the subsemiring generated by $\{0,1\}$, which is $\{0\} \cup \{n \cdot 1 \mid n \geq 1\}$. By the above identities 0 (resp., 1) is an additive (resp., multiplicative) identity in this subsemiring. □

As a consequence of Lemma 3.1 (point 4), one can check in time $O(n^2)$ for a semiring of size n whether it is $\{0,1\}$-free. We will not need this fact, since in our setting the semiring will be always fixed, i.e., not part of the input. Moreover, the class of all $\{0,1\}$-free semirings is closed

under taking subsemirings (this is trivial) and taking homomorphic images (by point 3). Finally, the class of $\{0,1\}$-free semirings is also closed under direct products. To see this, assume that $R \times R'$ is not $\{0,1\}$-free. Hence, there exists a subsemiring T of $R \times R'$ with an additive zero $(0,0')$ and a multiplicative one $(1,1') \neq (0,0')$. W.l.o.g. assume that $0 \neq 1$. Then the projection $\pi_1(T)$ onto the first component is a subsemiring of R, where 0 is an additive identity and $1 \neq 0$ is a multiplicative identity. By these remarks, the class of $\{0,1\}$-free finite semirings forms a pseudo-variety of finite semirings. Again, this fact will not be used in the rest of the paper, but it might be of independent interest.

4 Circuit evaluation and main results

We define circuits over general algebraic structures. Let $\mathcal{A} = (D, f_1, \ldots, f_k)$ be an algebraic structure. A *circuit* over $\mathcal{A}$ is a triple $\mathcal{C} = (V, A_0, \mathsf{rhs})$ where V is a finite set of *gates*, $A_0 \in V$ is the *output gate* and rhs (which stands for right-hand side) is a function that assigns to each gate $A \in V$ an element $a \in D$ or an expression of the form $f_i(A_1, \ldots, A_n)$, where $n = n_i$ and $A_1, \ldots, A_n \in V$ are called the *input gates for* A. Moreover, the binary relation $\{(A,B) \in V \times V \mid A \text{ is an input gate for } B\}$ is required to be acyclic. Its reflexive and transitive closure is a partial order on V that we denote with $\leq_{\mathcal{C}}$. Every gate A evaluates to an element $[A]_{\mathcal{C}} \in A$ in the natural way: If $\mathsf{rhs}(A) = a \in D$, then $[A]_{\mathcal{C}} = a$ and if $\mathsf{rhs}(A) = f_i(A_1, \ldots, A_n)$ then $[A]_{\mathcal{C}} = f_i([A_1]_{\mathcal{C}}, \ldots, [A_n]_{\mathcal{C}})$. Moreover, we define $[\mathcal{C}] = [A_0]_{\mathcal{C}}$ (the value computed by $\mathcal{C}$). If the circuit $\mathcal{C}$ is clear from the context, we also write $[A]$ instead of $[A]_{\mathcal{C}}$. We say that two circuits $\mathcal{C}_1$ and $\mathcal{C}_2$ over the structure $\mathcal{A}$ are *equivalent* if $[\mathcal{C}_1] = [\mathcal{C}_2]$. Sometimes we also use circuits without an output gate; such a circuit is just a pair (V, rhs). A subcircuit of $\mathcal{C}$ is the restriction of $\mathcal{C}$ to a downwards closed (w.r.t. $\leq_{\mathcal{C}}$) subset of V. A gate A with $\mathsf{rhs}(A) = f_i(A_1, \ldots, A_n)$ is called an *inner gate*, otherwise it is an *input gate* of $\mathcal{C}$. Quite often, we view a circuit as a directed acyclic graph, where the inner nodes are labelled with an operations f_i, and the leaf nodes are labelled with elements from D. In our proofs it is sometimes convenient to allow arbitrary terms built from $V \cup D$ using the operations $f_1, \ldots, f_k$ in right-hand sides. For instance, over a semiring $(R, +, \cdot)$ we might have $\mathsf{rhs}(A) = s \cdot B \cdot t + C + s$ for $s, t \in R$ and $B, C \in V$. A circuit is in *normal form*, if all right-hand sides are of the form $a \in D$ or $f_i(A_1, \ldots, A_n)$ with $A_1, \ldots, A_n \in V$. We will make use of the following simple fact:

Lemma 4.1. *A given circuit can be transformed in logspace into an equivalent normal form circuit.*

Proof. The only non-trivial part is the elimination of copy gates A with $\mathsf{rhs}(A) = B$ for a gate B; all other right-hand sides that violate the normal form have to be split up using fresh gates. This is easily done in logspace. For copy gates consider the directed graph G that contains for every copy gate A the gate A as well as the gate $\mathsf{rhs}(A)$. Moreover, there is a directed edge from A to $B = \mathsf{rhs}(A)$. This is a directed forest where the edges are oriented towards the roots since every node has at most one outgoing edge (and the graph is acyclic). By traversing all (deterministic) paths, we can compute the reflexive transitive closure G^* of G in logspace. Using G^* it is straightforward to eliminate copy gates: For every copy gate A we redefine $\mathsf{rhs}(A) = \mathsf{rhs}(B)$, where B is the unique node in G^* of outdegree zero such that (A, B) is an edge of G^*. □

The *circuit evaluation problem* $\mathsf{CEP}(\mathcal{A})$ for some algebraic structure $\mathcal{A}$ (say a semigroup or a semiring) is the following computational problem:

Input: A circuit $\mathcal{C}$ over $\mathcal{A}$ and an element $a \in D$ from its domain.
Output: Decide whether $[\mathcal{C}] = a$.

Note that for a finite structure $\mathcal{A}$, $\mathsf{CEP}(\mathcal{A})$ is basically equivalent to its computation variant, where one actually computes the output value $[\mathcal{C}]$ of the circuit: if $\mathsf{CEP}(\mathcal{A})$ belongs to a complexity class C, then the computation variant belongs to $\mathsf{AC}^0(\mathsf{C})$, and if the latter belongs to $\mathsf{AC}^0(\mathsf{C})$ then $\mathsf{CEP}(\mathcal{A})$ belongs to the decision fragment of $\mathsf{AC}^0(\mathsf{C})$.

Clearly, for every finite structure the circuit evaluation problem can be solved in polynomial time by evaluating all gates along the partial order $\leq_{\mathcal{C}}$. Ladner's classical P-completeness result for the Boolean circuit value problem [20] can be stated as follows:

Theorem 4.2 ([20]). *For the Boolean semiring $\mathbb{B}_2 = (\{0,1\}, \vee, \wedge)$, the problem $\mathsf{CEP}(\mathbb{B}_2)$ is P-complete.*

For semigroups, the following dichotomy was shown in [10]:

Theorem 4.3 ([10]). *Let S be a finite semigroup.*

- *If S is aperiodic, then $\mathsf{CEP}(S)$ is in NL.*
- *If S is solvable, then $\mathsf{CEP}(S)$ belongs to DET.*
- *If S is not solvable, then $\mathsf{CEP}(S)$ is P-complete.*

Some remarks should be made:

- In [10], Theorem 4.3 is only shown for monoids, but the extension to semigroups is straightforward: If the finite semigroup S has a non-solvable subgroup, then $\mathsf{CEP}(S)$ is P-complete, since the circuit evaluation problem for a non-solvable finite group is P-complete. On the other hand, if S is solvable (resp., aperiodic), then also the monoid S^1 is solvable (resp., aperiodic). This holds, since the subgroups of S^1 are exactly the subgroups of S together with $\{1\}$. Hence, $\mathsf{CEP}(S^1)$ is in DET (resp., NL), which implies that $\mathsf{CEP}(S)$ is in DET (resp., NL).
- In [10], the authors use the original definition $\mathsf{DET} = \mathsf{NC}^1(\mathsf{det})$ of Cook. But the arguments in [10] actually show that for a finite solvable semigroup, $\mathsf{CEP}(S)$ belongs to $\mathsf{AC}^0(\mathsf{det})$ (which is our definition of DET).
- In [10], the authors study two versions of the circuit evaluation problem for a semigroup S: What we call $\mathsf{CEP}(S)$ is called $\mathsf{UCEP}(S)$ (for "unrestricted circuit evaluation problem") in [10]. The problem $\mathsf{CEP}(S)$ is defined in [10] as the circuit evaluation problem, where in addition the input circuit must have the property that the output gate has no ingoing edges and all gates are reachable from the output gate. These conditions can be enforced with an $\mathsf{AC}^0(\mathsf{NL})$-precomputation. Hence, the difference between the two variants is only relevant for classes below NL. We only consider the unrestricted version of the circuit evaluation problem (where the input circuit is arbitrary). To keep notation simple, we decided to refer with CEP to the unrestricted version.

Let us fix a *finite* semiring $R = (R, +, \cdot)$ for the rest of the paper. Note that $\mathsf{CEP}(R_+)$ (resp., $\mathsf{CEP}(R_\bullet)$) is the restriction of $\mathsf{CEP}(R)$ to circuits without multiplication (resp., addition) gates. Since every commutative semigroup is solvable, Theorem 4.3 implies that $\mathsf{CEP}(R_+)$ belongs to DET. The main result of this paper is:

Theorem 4.4. *If the finite semiring R is $\{0,1\}$-free, then the problem $\mathsf{CEP}(R)$ belongs to the class $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+), \mathsf{CEP}(R_\bullet))$. Otherwise $\mathsf{CEP}(R)$ is P-complete.*

Note that if $\mathsf{CEP}(R_+)$ or $\mathsf{CEP}(R_\bullet)$ is NL-hard, then

$$\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+), \mathsf{CEP}(R_\bullet)) = \mathsf{AC}^0(\mathsf{CEP}(R_+), \mathsf{CEP}(R_\bullet)).$$

For example, this is the case, if R_+ or $R_\bullet$ is an aperiodic nontrivial monoid [10, Proposition 4.14].

The P-hardness statement in Theorem 4.4 is easy to show:

Lemma 4.5. *If the finite semiring R is not $\{0,1\}$-free, then the problem $\mathsf{CEP}(R)$ is P-complete.*

Proof. By Lemma 3.1, R contains either $\mathbb{B}_2$ or $\mathbb{Z}_d$ for some $d \geq 2$. In the former case, P-hardness follows from Ladner's theorem. Furthermore, one can reduce the P-complete Boolean circuit value problem over $\{0, 1, \wedge, \neg\}$ to $\mathsf{CEP}(\mathbb{Z}_d)$ for $d \geq 2$: A gate $z = x \wedge y$ is replaced by $z = x \cdot y$ and a gate $y = \neg x$ is replaced by $y = 1 + (d-1) \cdot x$. □

Theorem 4.3 and 4.4 yield the following corollaries:

Corollary 4.6. *Let R be a finite semiring.*

- *If R is not $\{0,1\}$-free or $R_\bullet$ is not solvable, then $\mathsf{CEP}(R)$ is P-complete.*
- *If R is $\{0,1\}$-free and $R_\bullet$ is solvable, then $\mathsf{CEP}(R)$ belongs to DET.*
- *If R is $\{0,1\}$-free and $R_\bullet$ is aperiodic, then $\mathsf{CEP}(R)$ belongs to NL.*

Let us present an application of Corollary 4.6.

Example 4.7. *An important semigroup construction found in the literature is the power construction. For a finite semigroup S one defines the* power semiring *$\mathcal{P}(S) = (2^S \setminus \{\emptyset\}, \cup, \cdot)$ with the multiplication $A \cdot B = \{ab \mid a \in A, b \in B\}$. Notice that if one includes the empty set, then the semiring would not be $\{0,1\}$-free: Take an idempotent $e \in S$. Then $\emptyset$ and $\{e\}$ form a copy of $\mathbb{B}_2$. Hence, the circuit evaluation problem is P-complete.*

Let us further assume that S is a monoid with identity 1 (the general case will be considered below). If S contains an idempotent $e \neq 1$ then also $\mathcal{P}(S)$ is not $\{0,1\}$-free: $\{e\}$ and $\{1, e\}$ form a copy of $\mathbb{B}_2$. On the other hand, if 1 is the unique idempotent of S, then S must be a group G. Assume that G is solvable; otherwise $\mathcal{P}(G)_\bullet$ is not solvable as well and has a P-complete circuit evaluation problem by Theorem 4.3. It is not hard to show that the subgroups of $\mathcal{P}(G)_\bullet$ correspond to the quotient groups of subgroups of G; see also [23]. Since G is solvable and the class of solvable groups is closed under taking subgroups and quotients, $\mathcal{P}(G)_\bullet$ is a solvable monoid. Moreover $\mathcal{P}(G)$ is $\{0,1\}$-free: Otherwise, Lemma 3.1 implies that there are non-empty subsets $A, B \subseteq G$ such that $A \neq B$, $A \cup B = B$ (and thus $A \subsetneq B$), $AB = BA = A^2 = A$, and $B^2 = B$. Hence, B is a subgroup of G and $A \subseteq B$. But then $B = AB = A$, which is a contradiction. By Corollary 4.6, $\mathsf{CEP}(\mathcal{P}(G))$ for a finite solvable group G belongs to DET.

Let us now classify the complexity of $\mathsf{CEP}(\mathcal{P}(S))$ for arbitrary semigroups S. A semigroup S is called a *local group* if for all $e \in E(S)$ the local monoid eSe is a group. It is known that in every finite local group S of size n the minimal semigroup ideal is $S^n = SE(S)S$, see [4, Proposition 2.3].

Theorem 4.8. *Let S be a finite semigroup. If S is a local group and solvable, then $\mathsf{CEP}(\mathcal{P}(S))$ belongs to DET. Otherwise $\mathsf{CEP}(\mathcal{P}(S))$ is P-complete.*

Proof. Let S be a finite local group which is solvable. By [7, Corollary 2.7] the multiplicative semigroup $\mathcal{P}(S)_\bullet$ is solvable as well. It remains to show that the semiring $\mathcal{P}(S)$ is $\{0,1\}$-free: Towards a contradiction assume that $\mathcal{P}(S)$ is not $\{0,1\}$-free. By Lemma 3.1, there exist non-empty sets $A \subsetneq B \subseteq S$ such that $AB = BA = A^2 = A$ and $B^2 = B$. Hence, B is a subsemigroup of S, which is also a local group, and A is a semigroup ideal in B. Since the minimal semigroup ideal of B is B^n for $n = |B|$ and $B^n = B$, we obtain $A = B$, which is a contradiction.

The case that S is not a local group follows from the arguments in Example 4.7. In that case, there exists a local monoid eSe which is not a group and hence contains an idempotent $f \neq e$. Since $\{\{f\}, \{e, f\}\}$ forms a copy of $\mathbb{B}_2$ it follows that $\mathsf{CEP}(\mathcal{P}(S))$ is P-complete. Finally, if S is not solvable, then also $\mathcal{P}(S)$ is not solvable and $\mathsf{CEP}(\mathcal{P}(S))$ is P-complete by Theorem 4.3. □

5 Proof of Theorem 4.4

The proof of Theorem 4.4 will proceed in two steps. In the first step we reduce the problem to evaluating circuits in which the computation respects a type-function defined in the following. In the second step, we show how to evaluate such circuits.

Definition 5.1. *Let $E = E(R)$ be the set of multiplicative idempotents. Let $\mathcal{C} = (V, \mathsf{rhs})$ be a circuit in normal form such that $[A]_{\mathcal{C}} \in ERE$ for all gates $A \in V$. A type-function for $\mathcal{C}$ is a mapping $\mathsf{type} : V \to E \times E$ such that the following conditions hold:*

- *For every $A \in V$ such that $\mathsf{type}(A) = (e, f)$ we have $[A]_{\mathcal{C}} \in eRf$.*
- *For every addition gate $A \in V$ with $\mathsf{rhs}(A) = B + C$ we have $\mathsf{type}(A) = \mathsf{type}(B) = \mathsf{type}(C)$.*
- *For every multiplication gate $A \in V$ with $\mathsf{rhs}(A) = B \cdot C$, $\mathsf{type}(B) = (e, e')$, and $\mathsf{type}(C) = (f', f)$ we have $\mathsf{type}(A) = (e, f)$.*

A circuit is called type admitting *if it admits a type-function.*

Note that we do not need an output gate in a type admitting circuit.

Definition 5.2. *A function $\alpha : R^m \to R$ ($m \geq 0$) is called* affine *if there are $a_1, b_1, \ldots, a_m, b_m, c \in R$ such that for all $x_1, \ldots, x_m \in R$:*

$$\alpha(x_1, \ldots, x_m) = \sum_{i=1}^{m} a_i x_i b_i + c \quad \text{or} \quad \alpha(x_1, \ldots, x_m) = \sum_{i=1}^{m} a_i x_i b_i.$$

We represent this affine function by the tuple $(a_1, b_1, \ldots, a_m, b_m, c)$ or $(a_1, b_1, \ldots, a_m, b_m)$.

Theorem 4.4 is now an immediate corollary of the following two propositions (and the obvious fact that an affine function with a constant number of inputs can be evaluated in AC^0).

Proposition 5.3. *Given a circuit $\mathcal{C}$ over the finite semiring R, one can compute the following data in $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+))$:*

- *an affine function $\alpha : R^m \to R$ for some $0 \leq m \leq |R|^4$,*
- *a type admitting circuit $\mathcal{C}' = (V', \mathsf{rhs}')$, and*
- *a list of gates $A_1, \ldots, A_m \in V'$ such that $[\mathcal{C}] = \alpha([A_1]_{\mathcal{C}'}, \ldots, [A_m]_{\mathcal{C}'})$.*

Proposition 5.4. *If R is $\{0,1\}$-free, then the restriction of $\mathsf{CEP}(R)$ to type admitting circuits is in $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+), \mathsf{CEP}(R_\bullet))$.*

It is not clear how to test efficiently whether a circuit is type admitting. But this is not a problem for us, since we will apply Proposition 5.4 only to circuits resulting from Proposition 5.3, which are type admitting by construction.

5.1 Step 1: Reduction to typing admitting circuits

In this section, we prove Proposition 5.3. Let $\mathcal{C}$ be a circuit in normal form over our fixed finite semiring $R = (R, +, \cdot)$ of size $n = |R|$. We assume that $n \geq 2$ (the case $n = 1$ is trivial). Throughout the section we will use $E = E(R)$. Note that $R^n = RER$ is closed under multiplication with elements from R. Thus, $\langle R^n \rangle$ is an ideal. Every element $a \in \langle R^n \rangle$ can be written as a finite sum $a = \sum_{i=1}^{k} a_i$ with $a_i \in R^n$. Moreover, since R is a fixed finite semiring, the number k of summands can be bounded by a constant that only depends on R.

The reduction to type admitting circuits is done in two steps:

$$\text{circuit over } R \xrightarrow{\text{Lemma 5.8}} \text{circuit over } \langle R^n \rangle = \langle RER \rangle \xrightarrow{\text{Lemma 5.9}} \text{type admitting circuit}$$

Before we prove Lemma 5.8 and 5.9, we prove a lemma that allows to eliminate certain input values from a circuit:

Lemma 5.5. *Assume that $I \subseteq R$ is a non-empty ideal of R. Let $\mathcal{C} = (V, A_0, \mathsf{rhs})$ be a circuit in normal form. Consider the set $U = \{A \in V \mid A \text{ is an inner gate or } \mathsf{rhs}(A) \in I\}$ and assume that $A_0 \in U$ and for all $A, B, C \in V$ the following holds:*

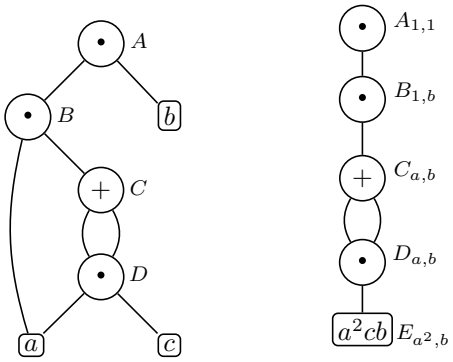

Figure 1: Illustration of the proof of Lemma 5.5 with the ideal $I = \{c\}$ in $S = \{a, b, c\}$.

- *If* $\mathsf{rhs}(A) = B \cdot C$ *then* $B \in U$ *or* $C \in U$.
- *If* $\mathsf{rhs}(A) = B + C$ *then* $B, C \in U$.

There is a logspace-computable function that returns for a given circuit $\mathcal{C}$ *with the above properties an equivalent circuit* $\mathcal{D}$ *in normal form over the ideal (and hence subsemiring)* I.

Proof. Let $U \subseteq V$ be defined as in the lemma. We first compute in logspace a circuit $\mathcal{C}' = (V', (A_0)_{1,1}, \mathsf{rhs}')$ which contains a gate $A_{\ell,r}$ for all gates $A \in U$ and $\ell, r \in R^1$ (recall that R^1 is R together with a fresh multiplicative identity 1) such that $[A_{\ell,r}]_{\mathcal{C}'} = \ell \cdot [A]_{\mathcal{C}} \cdot r$. Note that V' is non-empty since $A_0 \in U$. The gates of $\mathcal{C}'$ are indexed by elements of R^1 instead of R to simplify the notation in the following. To define the right-hand sides let us take a gate $A \in U$ and $\ell, r \in R^1$.

Case 1. $\mathsf{rhs}(A) = a \in I$. We set $\mathsf{rhs}'(A_{\ell,r}) = \ell a r$. Note that $\ell a r \in I$, since I is an ideal in R.

Case 2. $\mathsf{rhs}(A) = B + C$. Then we must have $B, C \in U$ and we set $\mathsf{rhs}'(A_{\ell,r}) = B_{\ell,r} + C_{\ell,r}$.

Case 3. $\mathsf{rhs}(A) = B \cdot C$. Then $B \in U$ or $C \in U$. If both B and C belong to U, then we set $\mathsf{rhs}'(A_{\ell,r}) = B_{\ell,1} \cdot C_{1,r}$. If $C \notin U$ then $B \in U$ and $\mathsf{rhs}(C) = c \in R \setminus I$. We set $\mathsf{rhs}'(A_{\ell,r}) = B_{\ell,cr}$. The case that $B \notin U$ is symmetric.

By Lemma 4.1 we can transform the circuit $\mathcal{C}'$ in logspace into a normal form circuit $\mathcal{D}$. The correctness of the above construction can be easily shown by induction along the partial order $\leq_{\mathcal{C}}$. □

Figure 1 illustrates the construction for $R = \{a, b, c\}$ and the ideal $I = \{c\}$ (the concrete semiring structure is not important). Note that $a^2cb \in I$. The circuit on the right-hand side is only the part of the constructed circuit that is rooted in the gate $A_{1,1}$. Copy gates are not eliminated.

A circuit $\mathcal{A} = (V, A_0, \mathsf{rhs})$ over the semiring $(\mathbb{N}, +, \cdot)$ is also called an *arithmetic circuit*. We further assume that all input gates of an arithmetic circuit have a right-hand side from $\{0, 1\}$. W.l.o.g. we can also assume that there are unique input gates with right-hand sides 0 and 1, respectively. The *multiplication depth* of a gate $A \in V$ is the maximal number of multiplication gates along every path from an input gate to A (where $\mathcal{A}$ is viewed as a directed acyclic graph). The multiplication depth of $\mathcal{A}$ is the maximal multiplication depth among all gates in $\mathcal{A}$. An *addition circuit* is an arithmetic circuit without multiplication gates. The following lemma is shown in [17, Lemma 6]:

Lemma 5.6. *Let* c *be a fixed constant. A given arithmetic circuit* $\mathcal{A}$ *of multiplication depth at most* c*, where in addition every multiplication gate is labelled with its multiplication depth, can be transformed in logspace into an addition circuit* $\mathcal{A}'$ *that contains every gate* A *of* $\mathcal{A}$ *and moreover satisfies* $[A]_{\mathcal{A}} = [A]_{\mathcal{A}'}$.

For the following proofs, it is sometimes more convenient to evaluate a circuit $\mathcal{C} = (V, A_0, \mathsf{rhs})$ over the free semiring $\mathbb{N}[R]$ generated by the set R.[1] Recall that this semiring consists of all mappings $f : R^+ \to \mathbb{N}$ with finite and non-empty support, where R^+ consists of all finite non-empty words over the alphabet R. So, there are two ways to evaluate $\mathcal{C}$: We can evaluate $\mathcal{C}$ over R (and this is our main interest) and we can evaluate $\mathcal{C}$ over $\mathbb{N}[R]$. In order to distinguish these two ways of evaluation, we write $[\![A]\!]_{\mathcal{C}} \in \mathbb{N}[R]$ for the value of gate $A \in V$ in $\mathcal{C}$, when $\mathcal{C}$ is evaluated in $\mathbb{N}[R]$. Again we omit the index $\mathcal{C}$ if it is clear from the context. Moreover, $[\![\mathcal{C}]\!] = [\![A_0]\!]_{\mathcal{C}}$. Note that $[\![A]\!]_{\mathcal{C}}$ is a mapping from R^+ to $\mathbb{N}$; hence for a word $w \in R^+$, $[\![A]\!]_{\mathcal{C}}(w)$ is a natural number. Let h be the canonical semiring homomorphism from $\mathbb{N}[R]$ to R that evaluates a non-commutative polynomial in the semiring R (the range of this mapping is the subsemiring generated by R). Thus, we have $[A]_{\mathcal{C}} = h([\![A]\!]_{\mathcal{C}})$ for every gate A and $[\mathcal{C}] = h([\![\mathcal{C}]\!])$. An example of a free evaluation of a circuit is shown in Figure 2 on the left.

Recall that $|R| = n \geq 2$. We define

$$R^{<n} = \{w \in R^+ \mid |w| < n\} \quad \text{and} \quad R^{\geq n} = \{w \in R^+ \mid |w| \geq n\}.$$

Note that these are sets of finite words over the alphabet R. So, these notations should not be confused with the notation R^n, which is a subset of R (the set of all n-fold products). In fact, we have $h(R^{\geq n}) = R^n = RER$. For every non-commutative polynomial $f \in \mathbb{N}[R]$ we define $f^\sigma, f^\lambda \in \mathbb{N}[R] \cup \{\bot\}$ (the short part and the long part of f) as follows ($\bot$ is a new symbol that stands for "undefined"):

1. If $\mathsf{supp}(f) \subseteq R^{<n}$, then $f^\sigma = f$ and $f^\lambda = \bot$.
2. If $\mathsf{supp}(f) \subseteq R^{\geq n}$, then $f^\sigma = \bot$ and $f^\lambda = f$.
3. Otherwise let $f^\sigma, f^\lambda \in \mathbb{N}[R]$ such that $f = f^\sigma + f^\lambda$, $\mathsf{supp}(f^\sigma) \subseteq R^{<n}$ and $\mathsf{supp}(f^\lambda) \subseteq R^{\geq n}$.

Note that either $f^\sigma \neq \bot$ or $f^\lambda \neq \bot$, and that the decomposition in 3 is unique. Moreover, if $f^\sigma \neq \bot \neq f^\lambda$, then $f = f^\sigma + f^\lambda$.

Example 5.7. *Let $R = \{a, b, c\}$ and thus $n = 3$. Let $f = 2abbca + 3caab + bab + 4ac + 7b \in \mathbb{N}[\{a, b, c\}]$. We have $f^\sigma = 4ac + 7b$ and $f^\lambda = 2abbca + 3caab + bab$.*

Lemma 5.8. *There is a function in $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+))$ that returns for a given circuit $\mathcal{C} = (V, A_0, \mathsf{rhs})$ either*

- *the semiring element $[\mathcal{C}] \in R$ (namely if $[\![\mathcal{C}]\!]^\lambda = \bot$), or*
- *a circuit $\mathcal{D}$ over the subsemiring $\langle R^n \rangle = \langle RER \rangle$ such that $[\mathcal{C}] = [\mathcal{D}]$ (namely if $[\![\mathcal{C}]\!]^\sigma = \bot$), or*
- *a circuit $\mathcal{D}$ over the subsemiring $\langle R^n \rangle = \langle RER \rangle$ and a semiring element $\sigma \in R$ such that $[\mathcal{C}] = [\mathcal{D}] + \sigma$ (namely if $[\![\mathcal{C}]\!]^\sigma \neq \bot \neq [\![\mathcal{C}]\!]^\lambda$).*

Proof. By Lemma 4.1 we can assume that $\mathcal{C}$ is in normal form. In the following, we omit the index $\mathcal{C}$ in $[A]_{\mathcal{C}}$ and $[\![A]\!]_{\mathcal{C}}$, where $A \in V$ is a gate of the circuit $\mathcal{C}$.

Step 1. We first compute in $\mathsf{AC}^0(\mathsf{NL})$ the set of all gates $A \in V$ such that $[\![A]\!]^\sigma \neq \bot$. For this, we construct in logspace an arithmetic circuit $\mathcal{A}$ over the semiring $(\mathbb{N}, +, \cdot)$ with gates A_w where $A \in V$ and $w \in R^{<n}$ such that $[A_w]_{\mathcal{A}} = [\![A]\!](w)$ as follows:

- If $\mathsf{rhs}(A) = a \in R$ then $\mathsf{rhs}(A_w) = \begin{cases} 1 & \text{if } w = a \\ 0 & \text{otherwise.} \end{cases}$
- If $\mathsf{rhs}(A) = B + C$ then $\mathsf{rhs}(A_w) = B_w + C_w$.

[1] Of course, there is no chance of efficiently evaluating a circuit over the free semiring $\mathbb{N}[R]$, since this might produce a doubly exponential number of monomials of exponential length. Circuit evaluation over $\mathbb{N}[R]$ is only used as a tool in our proofs.

- If $\mathsf{rhs}(A) = B \cdot C$ then $\mathsf{rhs}(A_w) = \sum_{w=uv} B_u \cdot C_v$, where the sum goes over all $u, v \in R^{<n}$ with $w = uv$.

Note that the empty sum is interpreted as 0 and that $\mathcal{A}$ has constant multiplication depth. Moreover, the multiplication depth of a gate A_w is $|w|-1$. By Lemma 5.6 we can transform in logspace $\mathcal{A}$ into an equivalent addition circuit, which we still denote with $\mathcal{A}$. The circuit $\mathcal{A}$ contains all gates A_w ($A \in V$, $w \in R^{<n}$) and possibly some additional gates.

We can assume that $\mathcal{A}$ has a unique input gate Z with right-hand side 1. Let U_σ be the set of all gates X of $\mathcal{A}$ such that $Z \leq_{\mathcal{A}} X$. These are exactly those gates of $\mathcal{A}$ that evaluate to a number larger than zero. Hence, for all $A \in V$ and $w \in R^{<n}$, we have $A_w \in U_\sigma$ if and only if $[\![A]\!](w) > 0$. Moreover, $[\![A]\!]^\sigma \neq \bot$ if and only if $A_w \in U_\sigma$ for some $w \in R^{<n}$. The set U_σ can be computed in $\mathsf{AC}^0(\mathsf{NL})$. Hence, we can also compute for every $A \in V$ the information whether $[\![A]\!]^\sigma \neq \bot$ and, in case $[\![A]\!]^\sigma \neq \bot$, the set $\mathsf{supp}([\![A]\!]^\sigma) = \mathsf{supp}([\![A]\!]) \cap R^{<n}$.

Step 2. For each gate $A \in V$ with $[\![A]\!]^\sigma \neq \bot$ we now compute the semiring element $h([\![A]\!]^\sigma) \in R$. For this we construct in logspace a circuit over R_+ that evaluates to $h([\![A]\!]^\sigma)$. Hence, $h([\![A]\!]^\sigma)$ can be computed using oracle access to $\mathsf{CEP}(R_+)$.

We first remove from the arithmetic addition circuit $\mathcal{A}$ all gates that are not in U_σ. This may produce copy gates (which is not a problem). Moreover, gate Z is now the only input gate of $\mathcal{A}$. For a semiring element $a \in R$ we define the circuit $\mathcal{C}_a$ (over R_+) by taking the addition circuit $\mathcal{A}$ and redefining $\mathsf{rhs}_\sigma(Z) = a$. Then, for every gate $A_w \in U_\sigma$ ($A \in V$, $w \in R^{<n}$) we have $[A_w]_{\mathcal{C}_a} = [\![A]\!](w) \cdot a$. In particular, if $[\![A]\!]^\sigma \neq \bot$, then

$$h([\![A]\!]^\sigma) = \sum_{w \in \mathsf{supp}([\![A]\!]) \cap R^{<n}} [\![A]\!](w) \cdot h(w) \;=\; \sum_{w \in \mathsf{supp}([\![A]\!]) \cap R^{<n}} [A_w]_{\mathcal{C}_{h(w)}}.$$

From the circuits $\mathcal{C}_{h(w)}$ we can construct in logspace a circuit over R_+ for this semiring element. Evaluating this circuit using oracle access to $\mathsf{CEP}(R_+)$ yields the element $h([\![A]\!]^\sigma)$.

Step 3. Next, we compute in $\mathsf{AC}^0(\mathsf{NL})$ the set of all gates $A \in V$ such that $[\![A]\!]^\lambda \neq \bot$. Since $n \geq 2$ (our initial assumption on the semiring R), we have $[\![A]\!]^\lambda \neq \bot$ if and only if there exist a gate $A' \leq_{\mathcal{C}} A$ with $\mathsf{rhs}(A') = B \cdot C$ and words $w_1, w_2 \in R^{<n}$ such that $|w_1 w_2| \geq n$, $[\![B]\!](w_1) > 0$ (i.e., $B_{w_1} \in U_\sigma$) and $[\![C]\!](w_2) > 0$ (i.e., $C_{w_2} \in U_\sigma$). This condition can be tested in NL. Hence, we can assume that the set of all $A \in V$ with $[\![A]\!]^\lambda \neq \bot$ is computed. If $[\![A_0]\!]^\lambda = \bot$, then we must have $[\![A_0]\!]^\sigma \neq \bot$ and we return the previously computed semiring element $h([\![A_0]\!]^\sigma)$, which is $[\mathcal{C}]$ in this case. Let us now assume that $[\![A_0]\!]^\lambda \neq \bot$.

Step 4. We then construct a circuit $\mathcal{C}_\lambda = (V_\lambda, (A_0)_\lambda, \mathsf{rhs}_\lambda)$, which contains for every gate $A \in V$ with $[\![A]\!]^\lambda \neq \bot$ a gate A_λ such that $[A_\lambda]_{\mathcal{C}_\lambda} = h([\![A]\!]^\lambda)$. In particular, $[\mathcal{C}_\lambda] = h([\![\mathcal{C}]\!]^\lambda)$.

In a first step, we compute in AC_0 the set M^λ of all multiplication gates $A \in V$ such that the following conditions hold: $[\![A]\!]^\lambda \neq \bot$, $\mathsf{rhs}(A) = B \cdot C$ for $B, C \in V$, $[\![B]\!]^\sigma \neq \bot \neq [\![C]\!]^\sigma$, and there exist $u \in \mathsf{supp}([\![B]\!]^\sigma)$, $v \in \mathsf{supp}([\![C]\!]^\sigma)$ with $|uv| \geq n$. This means that in the product $[\![B]\!] \cdot [\![C]\!]$ a monomial of length at least n arises from monomials $u \in \mathsf{supp}([\![B]\!])$, $v \in \mathsf{supp}([\![C]\!])$, both of which have length smaller than n.

Next, for every multiplication gate $A \in M^\lambda$, where $\mathsf{rhs}(A) = B \cdot C$ we compute in $\mathsf{AC}_0(\mathsf{CEP}(R_+))$ the semiring element

$$m_A := \sum_{u,v} ([\![B]\!](u)[\![C]\!](v)) \cdot h(uv) \in \langle R^n \rangle,$$

where the sum is taken over all words $u \in \mathsf{supp}([\![B]\!]^\sigma)$, $v \in \mathsf{supp}([\![C]\!]^\sigma)$ with $|uv| \geq n$. Let us take the addition circuit $\mathcal{A}$ constructed in Step 1 and 2 above. We add to $\mathcal{A}$ a single layer of multiplication gates $A_{u,v}$, where $B_u, C_v \in U_\sigma$. The right-hand side of $A_{u,v}$ is $B_u \cdot C_v$. Using Lemma 5.6, we can transform this circuit in logspace into an equivalent addition circuit; let us denote this circuit with $\mathcal{A}^2$. Clearly, gate $A_{u,v}$ evaluates to the number $[\![B]\!](u)[\![C]\!](v) \in \mathbb{N}$. This number is larger than zero, since $B_u, C_v \in U_\sigma$. Hence, by replacing in the addition circuit $\mathcal{A}^2$ the input value 1 by the semigroup element $h(uv)$, we obtain a circuit over the semigroup R_+, which

we can evaluate using oracle access to $\mathsf{CEP}(R_+)$. The value of gate $A_{u,v}$ yields the semiring element $([\![B]\!](u)[\![C]\!](v)) \cdot h(uv)$. Finally, the sum of all these values (for all $u \in \mathsf{supp}([\![B]\!]^\sigma)$, $v \in \mathsf{supp}([\![C]\!]^\sigma)$ with $|uv| \geq n$) can be computed by another AC^0-computation (it is a sum of a constant number of semiring elements).

It remains to define the right-hand sides of the gates A_λ in $\mathcal{C}_\lambda$. We distinguish the following cases (note that A must be an inner gate of $\mathcal{C}$ since $n \geq 2$ and $\mathcal{C}$ is in normal form).

Case 1. $\mathsf{rhs}(A) = B + C$. Then we must have $[\![B]\!]^\lambda \neq \bot$ or $[\![C]\!]^\lambda \neq \bot$ (otherwise $[\![A]\!]^\lambda = \bot$) and we set

$$\mathsf{rhs}_\lambda(A_\lambda) = \begin{cases} B_\lambda, & \text{if } [\![C]\!]^\lambda = \bot, \\ C_\lambda, & \text{if } [\![B]\!]^\lambda = \bot, \\ B_\lambda + C_\lambda, & \text{otherwise.} \end{cases}$$

Case 2. $\mathsf{rhs}(A) = B \cdot C$, $A \in M^\lambda$, and $\bot \notin \{[\![B]\!]^\sigma, [\![B]\!]^\lambda, [\![C]\!]^\sigma, [\![C]\!]^\lambda\}$. Then we set

$$\mathsf{rhs}_\lambda(A_\lambda) = B_\lambda \cdot C_\lambda + h([\![B]\!]^\sigma) \cdot C_\lambda + B_\lambda \cdot h([\![C]\!]^\sigma) + m_A. \tag{1}$$

If $A \notin M^\lambda$ but $\bot \notin \{[\![B]\!]^\sigma, [\![B]\!]^\lambda, [\![C]\!]^\sigma, [\![C]\!]^\lambda\}$ then we take the same definition but omit the summand m_A.

Let us explain the definition (1). We have

$$\begin{aligned} [\![A]\!]^\sigma + [\![A]\!]^\lambda &= [\![A]\!] \\ &= [\![B]\!] \cdot [\![C]\!] = ([\![B]\!]^\sigma + [\![B]\!]^\lambda) \cdot ([\![C]\!]^\sigma + [\![C]\!]^\lambda) \\ &= [\![B]\!]^\lambda \cdot [\![C]\!]^\lambda + [\![B]\!]^\sigma \cdot [\![C]\!]^\lambda + [\![B]\!]^\lambda \cdot [\![C]\!]^\sigma + [\![B]\!]^\sigma \cdot [\![C]\!]^\sigma. \end{aligned}$$

By selecting from the last line all monomials of length at least n, we get

$$[\![A]\!]^\lambda = [\![B]\!]^\lambda \cdot [\![C]\!]^\lambda + [\![B]\!]^\sigma \cdot [\![C]\!]^\lambda + [\![B]\!]^\lambda \cdot [\![C]\!]^\sigma + m_A.$$

Applying to this equality the morphism h and noting that $BC_{u,v}^{h(uv)}$ evaluates to $([\![B]\!](u)[\![C]\!](v)) \cdot h(uv)$ shows that (1) is indeed the right definition for $\mathsf{rhs}_\lambda(A_\lambda)$.

Case 3. $\mathsf{rhs}(A) = B \cdot C$ and $\bot \in \{[\![B]\!]^\sigma, [\![B]\!]^\lambda, [\![C]\!]^\sigma, [\![C]\!]^\lambda\}$. Then the corresponding terms on the right-hand side of (1) are omitted. More precisely if $[\![X]\!]^\sigma = \bot$ ($X \in \{B, C\}$) then we omit in (1) the product involving $h([\![X]\!]^\sigma)$ as well as the semiring element m_A, and if $[\![X]\!]^\lambda = \bot$ ($X \in \{B, C\}$) then we omit in (1) the two products involving X_λ. The reader may also interpret $\bot$ as zero and then do the obvious simplifications in (1) (but note that the semiring $\mathbb{N}[R]$ has no additive zero element). For example, if $[\![B]\!]^\lambda = [\![C]\!]^\sigma = \bot$ and $[\![B]\!]^\sigma \neq \bot \neq [\![C]\!]^\lambda$ then $\mathsf{rhs}_\lambda(A_\lambda) = h([\![B]\!]^\sigma)C_\lambda$. Since $[\![A]\!]^\lambda \neq \bot$, at least one of the summands in (1) remains.

Figure 2 shows an example of the above construction of the circuit $\mathcal{C}^\lambda$, where $n = 2$. The shaded parts are those parts that are removed because they would yield zero terms. These are exactly those parts of the right-hand sides that are removed in the above Case 3. To the right of each gate X, the value $[\![X]\!]^\sigma$ is written. Note that the output gate evaluates to $2aab + 2ab$ which is indeed $(2aab + 2ab + 2a)^\lambda$.

Step 5. We now apply Lemma 4.1 and transform in logspace $\mathcal{C}_\lambda$ into a circuit $\mathcal{C}'$ in normal form. We have to argue that the circuit $\mathcal{C}'$ satisfies the conditions from Lemma 5.5 for the ideal $I = \langle R^n \rangle$. The input values of the circuit $\mathcal{C}_\lambda$ are elements from $\langle R^n \rangle$ (they occur as the m_A in (1)) and the $h([\![X]\!]^\sigma)$ for $X \in V$. Only the input values $h([\![X]\!]^\sigma)$ can belong to $R \setminus \langle R^n \rangle$ (they can also belong to $\langle R^n \rangle$). The circuit $\mathcal{C}'$ is obtained from $\mathcal{C}_\lambda$ by (i) eliminating copy gates (that may arise from the above Case 1) and (ii) splitting up right-hand sides of the form (1) (or a simpler form, see Case 3). Note that in the circuit $\mathcal{C}'$ an input gate Z with $\mathsf{rhs}_{\mathcal{C}'}(Z) \in R \setminus \langle R^n \rangle$ can only occur in right-hand sides of multiplication gates. Such a right-hand side must be of the form $B_\lambda \cdot Z$ or $Z \cdot C_\lambda$ (which is obtained from splitting up the expression in (1)). Here, B_λ and C_λ are gates from the circuit $\mathcal{C}_\lambda$.

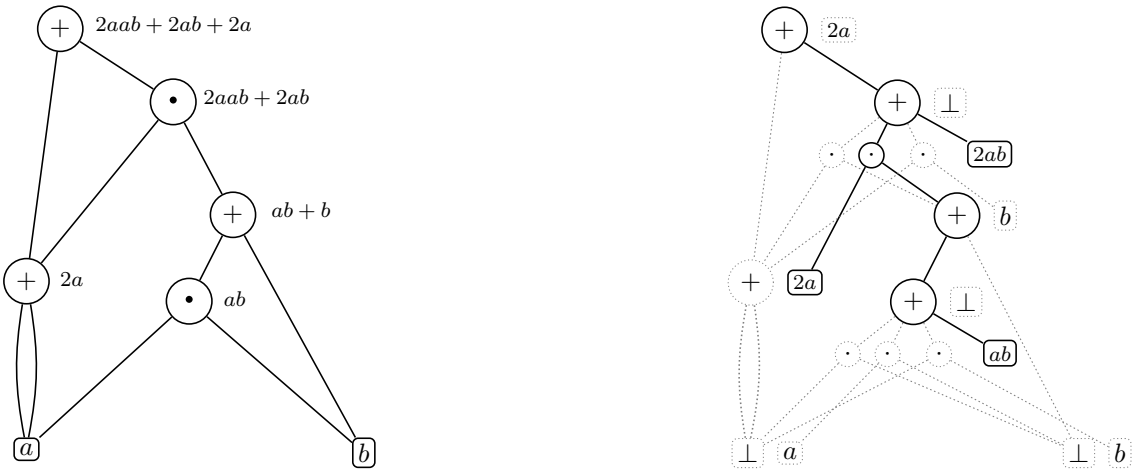

Figure 2: The construction of the circuit $\mathcal{C}_\lambda$ in the proof of Lemma 5.8.

But a gate A_λ of the circuit $\mathcal{C}_\lambda$ cannot be transformed into an input gate of $\mathcal{C}'$ with a right-hand side from $R \setminus \langle R^n \rangle$ (note that the values $h([\![X]\!]^\sigma)$ are "guarded" in (1) by multiplications with gates Y_λ). This shows that the conditions for Lemma 5.5 are satisfied.

Finally, Lemma 5.5 allows us to transform in logspace the circuit $\mathcal{C}'$ into an equivalent normal form circuit $\mathcal{D}$ over the subsemiring $\langle R^n \rangle = \langle RER \rangle$. This circuit $\mathcal{D}$ satisfies $[\mathcal{D}] = h([\![\mathcal{C}]\!]^\lambda)$. We output this circuit together with the previously computed value $h([\![\mathcal{C}]\!]^\sigma)$ (if this value is not $\bot$). Then the output specification from the lemma is satisfied. □

The next lemma transforms a normal form circuit over $\langle RER \rangle$ into a type admitting circuit.

Lemma 5.9. *Given a normal form circuit $\mathcal{C} = (V, A_0, \mathsf{rhs})$ over $\langle RER \rangle$, one can compute in $\mathsf{AC}^0(\mathsf{NL})$:*

- *a type admitting circuit $\mathcal{C}' = (V', \mathsf{rhs}')$ (without output gate),*
- *a non-empty list of distinguished gates $A_1, \ldots, A_m \in V'$, where $m \leq |R|^4$, and*
- *elements $\ell_1, r_1, \ldots, \ell_m, r_m \in R$ such that $[\mathcal{C}] = \sum_{i=1}^m \ell_i [A_i]_{\mathcal{C}'} r_i$.*

Proof. Let us interpret the circuit $\mathcal{C} = (V, A_0, \mathsf{rhs})$ over the free semiring $\mathbb{N}[R]$. For each input gate A we can write $[A]$ as $\sum_{i=1}^k s_i e_i^3 t_i$ for a constant k (that only depends on R), $s_i, t_i \in R$, $e_i \in E$ and redefine $\mathsf{rhs}(A) = \sum_{i=1}^k s_i e_i^3 t_i$ (a sum of k monomials of length 5). Thus for all $A \in V$ we have $\mathsf{supp}([\![A]\!]_{\mathcal{C}}) \subseteq (RE^3R)^+ \subseteq RER^*ER$.

Let us define for every inner gate A of $\mathcal{C}$ the set

$$P_A = \{(s, e, f, t) \in R \times E \times E \times R \mid \mathsf{supp}([\![A]\!]_{\mathcal{C}}) \cap seR^*ft \neq \emptyset\}.$$

Hence, $|P_A| \leq |R|^4$. We claim that the sets P_A can be computed in $\mathsf{AC}^0(\mathsf{NL})$. For this, note that $(s, e, f, t) \in P_A$ if and only if (i) $e = f$ and there exists an input gate C of $\mathcal{C}$ such that the following conditions hold:

- $\mathsf{rhs}(C)$ contains the monomial se^3t.
- There is a path from C to A (possible the empty path) where all gates are addition gates.

or (ii) there exist input gates C_1, C_2 of $\mathcal{C}$, and a multiplication gate B such that the following conditions hold:

- $\mathsf{rhs}(C_1)$ contains the monomial se^3t' for some $t' \in R$.

- $\mathsf{rhs}(C_2)$ contains the monomial $s'f^3t$ for some $s' \in R$.
- There is a path from C_1 to B such that for every edge (X, Y) along this path, where Y is a multiplication gate, $\mathsf{rhs}(Y) = X \cdot Z$ for some gate Z.
- There is a path from C_2 to B such that for every edge (X, Y) along this path, where Y is a multiplication gate, $\mathsf{rhs}(Y) = Z \cdot X$ for some gate Z.
- There is a path from B to A (possible the empty path) where except for B all gates are addition gates.

These conditions can be checked in NL.

We next compute in logspace a new circuit $\mathcal{D}$ that contains for every gate A of $\mathcal{C}$ and every tuple $(s, e, f, t) \in P_A$ a gate $A_{s,e,f,t}$ such that the following holds, where as usual $[\![A_{s,e,f,t}]\!]_{\mathcal{D}}$ denotes the evaluation of the gate $A_{s,e,f,t}$ in the free semiring $\mathbb{N}[R]$ and $L(A, s, e, f, t) := \mathsf{supp}([\![A]\!]_{\mathcal{C}}) \cap seR^*ft$ (which is non-empty since $(s, e, f, t) \in P_A$):

$$[\![A_{s,e,f,t}]\!]_{\mathcal{D}} = \sum_{w \in L(A,s,e,f,t)} [\![A]\!]_{\mathcal{C}}(w) \cdot w \tag{2}$$

Intuitively, we decompose the polynomial $[\![A]\!]_{\mathcal{C}}$ into several summands according to the first two and last two symbols in every monomial. We define the rules of $\mathcal{D}$ as follows, where A is a gate of $\mathcal{C}$:

Case 1. $\mathsf{rhs}(A) = \sum_{i=1}^{k} s_i e_i^3 t_i$. Then, we have $P_A = \{(s_i, e_i, e_i, t_i) \mid 1 \leq i \leq k\}$ and we set

$$\mathsf{rhs}(A_{s_i,e_i,e_i,t_i}) = s_i e_i^3 t_i.$$

Case 2. $\mathsf{rhs}(A) = B \cdot C$ and $(s, e, f, t) \in P_A$. We set

$$\mathsf{rhs}(A_{s,e,f,t}) = \sum_{(s,e,f',t') \in P_B} \sum_{(s',e',f,t) \in P_C} B_{s,e,f',t'} \cdot C_{s',e',f,t}.$$

Case 3. $\mathsf{rhs}(A) = B + C$ and $(s, e, f, t) \in P_A$. We set

$$\mathsf{rhs}(A_{s,e,f,t}) = B_{s,e,f,t} + C_{s,e,f,t}.$$

With these right-hand sides, property (2) is easy to verify.

The idea of the last step is the following: Let $\overline{u} = (s, e, f, t) \in P_A$. Every non-commutative polynomial $[\![A_{\overline{u}}]\!]_{\mathcal{D}}$ has the property that each of its monomials starts with see and ends with fft. By factoring out the common prefix se and suffix ft, respectively, we can write $[\![A_{\overline{u}}]\!]_{\mathcal{D}} = segft$, where $g \in e\,\mathbb{N}[R]f$ or $g = e$ (the latter case occurs if A is an input gate with right-hand side se^3t, in which case we have $e = f$). We now construct in logspace a circuit $\mathcal{C}'$, which contains gates $A'_{s,e,f,t}$ (where $A_{s,e,f,t}$ is a gate of $\mathcal{D}$ as above) such that in the free semiring $\mathbb{N}[R]$, $A'_{s,e,f,t}$ evaluates to the above polynomial g. We define the right-hand side of $A'_{s,e,f,t}$ again by a case distinction, where we use the right-hand sides for $\mathcal{D}$ that we defined in the Cases 1-3 above.

Case 1. $e = f$ and $\mathsf{rhs}(A_{s,e,e,t}) = se^3t$. Then, we set $\mathsf{rhs}(A'_{s,e,e,t}) = e$.

Case 2. $\mathsf{rhs}(A_{s,e,f,t}) = \sum_{(s,e,f',t') \in P_B} \sum_{(s',e',f,t) \in P_C} B_{s,e,f',t'} \cdot C_{s',e',f,t}$. We set

$$\mathsf{rhs}(A'_{s,e,f,t}) = \sum_{(s,e,f',t') \in P_B} \sum_{(s',e',f,t) \in P_C} B'_{s,e,f',t'}(f't's'e')C_{s',e',f,t}. \tag{3}$$

Case 3. $\mathsf{rhs}(A_{s,e,f,t}) = B_{s,e,f,t} + C_{s,e,f,t}$. We set

$$\mathsf{rhs}(A'_{s,e,f,t}) = B'_{s,e,f,t} + C'_{s,e,f,t}.$$

It is now straightforward to verify that for every gate A of $\mathcal{C}$ we have:

$$[\![A]\!]_{\mathcal{C}} = \sum_{(s,e,f,t)\in P_A} [\![A_{s,e,f,t}]\!]_{\mathcal{D}} = \sum_{(s,e,f,t)\in P_A} se[\![A'_{s,e,f,t}]\!]_{\mathcal{C}'} ft$$

Hence, if we evaluate the circuits $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{C}'$ in the semiring R we get

$$[A]_{\mathcal{C}} = \sum_{(s,e,f,t)\in P_A} [A_{s,e,f,t}]_{\mathcal{D}} \quad = \quad \sum_{(s,e,f,t)\in P_A} se[A'_{s,e,f,t}]_{\mathcal{C}'} ft.$$

Note that $[A'_{s,e,f,t}]_{\mathcal{C}'} \in eRf$, which holds, since $[A'_{s,e,f,t}]_{\mathcal{C}'} = h([\![A'_{s,e,f,t}]\!]_{\mathcal{C}'})$ and every monomial of $[\![A'_{s,e,f,t}]\!]_{\mathcal{C}'}$ starts with e and ends with f. Moreover, every input value of the circuit $\mathcal{C}'$ is from ERE: These are the elements e (in Case 1) and $f't's'e'$ (in Case 2).

Note that $\mathcal{C}'$ admits a type function; We set $\mathsf{type}(A'_{s,e,f,t}) = (e, f)$. Moreover, using Lemma 4.1 we transform $\mathcal{C}'$ in logspace into normal form by splitting up right-hand sides of the form (3). Thereby we extend the type-mapping to the new gates that are introduced. For instance, if we introduce a gate with right-hand side $B'_{s,e,f',t'}(f't's'e')$ (which occurs in (3)), then this gate gets the type (e, e'), and the gate that computes (in two steps) $B'_{s,e,f',t'}(f't's'e')C_{s',e',f,t}$ gets the type (e, f). This ensures that the three conditions from Definition 5.1 are satisfied. □

Combining Lemma 5.8 and 5.9 immediately yields Proposition 5.3.

5.2 Step 2: A parallel evaluation algorithm for type admitting circuits

In this section we prove Proposition 5.4. We present a parallel evaluation algorithm for type admitting circuits. This algorithm terminates after at most $|R|$ rounds, if R has a so called rank-function, which we define first. As before, let $E = E(R)$.

Definition 5.10. *We call a function* $\mathsf{rank} : R \to \mathbb{N} \setminus \{0\}$ *a* rank-function *for R if it satisfies the following conditions for all $a, b \in R$:*

1. $\mathsf{rank}(a) \leq \mathsf{rank}(a + b)$
2. $\mathsf{rank}(a), \mathsf{rank}(b) \leq \mathsf{rank}(a \cdot b)$
3. *If $a, b \in eRf$ for some $e, f \in E$ and $\mathsf{rank}(a) = \mathsf{rank}(a + b)$, then $a = a + b$.*

Note that if $R_\bullet$ is a monoid, then one can choose $e = 1 = f$ in the third condition in Definition 5.10, which is therefore equivalent to: If $\mathsf{rank}(a) = \mathsf{rank}(a + b)$ for $a, b \in R$, then $a = a + b$.

Example 5.11 (Example 4.7 continued)**.** *Let G be a finite group and consider the semiring $\mathcal{P}(G)$. One can verify that the function $A \mapsto |A|$, where $\emptyset \neq A \subseteq G$, is a rank-function for $\mathcal{P}(G)$. On the other hand, if S is a finite semigroup, which is not a group, then S cannot be cancellative. Assume that $ab = ac$ for $a, b, c \in S$ with $b \neq c$. Then $\{a\} \cdot \{b, c\} = \{ab\}$. This shows that the function $A \mapsto |A|$ is not a rank-function for $\mathcal{P}(S)$.*

Theorem 5.12. *If the finite semiring R has a rank-function* rank*, then the restriction of* $\mathsf{CEP}(R)$ *to type admitting circuits belongs to* $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+), \mathsf{CEP}(R_\bullet))$.

Proof. Let $\mathcal{C} = (V, A_0, \mathsf{rhs})$ be a circuit with the type function type. We present an algorithm which partially evaluates the circuit in a constant number of phases, where each phase can be carried out in $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+), \mathsf{CEP}(R_\bullet))$ and the following invariant is preserved:

Invariant. *After phase k all gates A with $\mathsf{rank}([A]_{\mathcal{C}}) \leq k$ are evaluated, i.e., are input gates.*

In the beginning, i.e., for $k = 0$, the invariant clearly holds (since 0 is not in the range of the rank-function). After $\mathsf{max}\{\mathsf{rank}(a) \mid a \in R\}$ (which is a constant) many phases the output gate A_0 is evaluated. We present phase k of the algorithm, assuming that the invariant holds after phase

$k-1$. Thus, all gates A with $\mathsf{rank}([A]_{\mathcal{C}}) < k$ of the current circuit $\mathcal{C}$ are input gates. The goal of phase k is to evaluate all gates A with $\mathsf{rank}([A]_{\mathcal{C}}) = k$. For this, we proceed in two steps:

Step 1. As a first step the algorithm evaluates all subcircuits that only contain addition and input gates. This maintains the invariant and is possible in $\mathsf{AC}^0(\mathsf{NL}, \mathsf{CEP}(R_+))$. After this step, every addition-gate A has at least one inner input gate, which we denote by $\mathsf{inner}(A)$ (if both input gates are inner gates, then choose one arbitrarily). The NL-oracle access is needed to compute the set of all gates A for which no multiplication gate $B \leq_{\mathcal{C}} A$ exists.

Step 2. Define the multiplicative circuit $\mathcal{C}' = (V, A_0, \mathsf{rhs}')$ by

$$\mathsf{rhs}'(A) = \begin{cases} \mathsf{inner}(A) & \text{if } A \text{ is an addition-gate,} \\ \mathsf{rhs}(A) & \text{if } A \text{ is a multiplication gate or input gate.} \end{cases} \tag{4}$$

The circuit $\mathcal{C}'$ can be brought into normal form by Lemma 4.1 and then evaluated using the oracle for $\mathsf{CEP}(R_\bullet)$. A gate $A \in V$ is called *locally correct* if (i) A is an input gate or multiplication gate of $\mathcal{C}$, or (ii) A is an addition gate of $\mathcal{C}$ with $\mathsf{rhs}(A) = B + C$ and $[A]_{\mathcal{C}'} = [B]_{\mathcal{C}'} + [C]_{\mathcal{C}'}$. We compute the set

$$W = \{A \in V \mid B \text{ is locally correct for all gates } B \text{ with } B \leq_{\mathcal{C}} A\}$$

in $\mathsf{AC}^0(\mathsf{NL})$. A simple induction shows that for all $A \in W$ we have $[A]_{\mathcal{C}} = [A]_{\mathcal{C}'}$. Hence we can set $\mathsf{rhs}(A) = [A]_{\mathcal{C}'}$ for all $A \in W$. This concludes phase k of the algorithm.

To prove that the invariant still holds after phase k, we show that for each gate $A \in V$ with $\mathsf{rank}([A]_{\mathcal{C}}) \leq k$ we have $A \in W$. This is shown by induction over the depth of A in $\mathcal{C}$. Assume that $\mathsf{rank}([A]_{\mathcal{C}}) \leq k$. By the first two conditions from Definition 5.10, all gates $B <_{\mathcal{C}} A$ satisfy $\mathsf{rank}([B]_{\mathcal{C}}) \leq k$. Thus, the induction hypothesis yields $B \in W$ and hence $[B]_{\mathcal{C}} = [B]_{\mathcal{C}'}$ for all gates $B <_{\mathcal{C}} A$.

It remains to show that A is locally correct, which is clear if A is an input gate or a multiplication gate. So assume that $\mathsf{rhs}(A) = B + C$ where $B = \mathsf{inner}(A)$, which implies $[A]_{\mathcal{C}'} = [B]_{\mathcal{C}'}$ by (4). Since B is an inner gate, which is not evaluated after phase $k-1$, it holds that $\mathsf{rank}([B]_{\mathcal{C}}) \geq k$ and therefore $\mathsf{rank}([A]_{\mathcal{C}}) = \mathsf{rank}([B]_{\mathcal{C}}) = k$. By Definition 5.1 there exist idempotents $e, f \in E$ with $\mathsf{type}(B) = \mathsf{type}(C) = (e, f)$ and thus $[B]_{\mathcal{C}}, [C]_{\mathcal{C}} \in eRf$. The third condition from Definition 5.10 implies that $[A]_{\mathcal{C}} = [B]_{\mathcal{C}} + [C]_{\mathcal{C}} = [B]_{\mathcal{C}}$. We get

$$[A]_{\mathcal{C}'} = [B]_{\mathcal{C}'} = [B]_{\mathcal{C}} = [A]_{\mathcal{C}} = [B]_{\mathcal{C}} + [C]_{\mathcal{C}} = [B]_{\mathcal{C}'} + [C]_{\mathcal{C}'}.$$

Therefore A is locally correct. □

Example 5.13 (Example 4.7 continued)**.** *Figure 3 shows a circuit $\mathcal{C}$ over the power semiring $\mathcal{P}(G)$ of the group $G = (\mathbb{Z}_5, +)$. Recall from Example 5.11 that the function $A \mapsto |A|$ is a rank function for $\mathcal{P}(G)$. We illustrate one phase of the algorithm. All gates A with $\mathsf{rank}([A]) < 3$ are evaluated in the circuit $\mathcal{C}$ shown in (a). The goal is to evaluate all gates A with $\mathsf{rank}([A]) = 3$. The first step would be to evaluate maximal $\cup$-circuits, which is already done. In the second step the circuit $\mathcal{C}'$ (shown in (b)) from the proof of Lemma 5.12 is computed and evaluated using the oracle for $\mathsf{CEP}(\mathbb{Z}_5, +)$. The dotted wires do not belong to the circuit $\mathcal{C}'$. All locally correct gates are shaded. Note that the output gate is locally correct but its right child is not locally correct. All other shaded gates form a downwards closed set, which is the set W from the proof. These gates can be evaluated such that in the resulting circuit (shown in (c)) all gates which evaluate to elements of rank 3 are evaluated.*

For the proof of Proposition 5.4, it remains to show that every finite $\{0,1\}$-free semiring has a rank-function.

Lemma 5.14. *Let R be $\{0,1\}$-free. Let $e, f \in R$ such that*

- $ef = fe = f$,

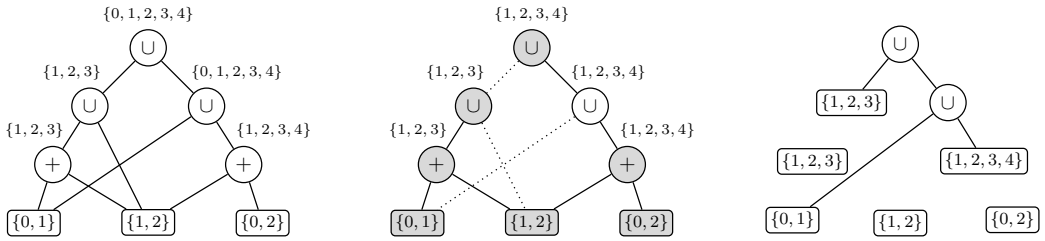

Figure 3: The parallel evaluation algorithm over the power semiring $\mathcal{P}(\mathbb{Z}_5)$.

- $e^2 = e$,
- $f^2 = f + f = f$.

Then $e + f = f$.

Proof. With $f = 0$ and $e + f = 1$ all equations from Lemma 3.1 (point 4) hold. Hence, we must have $e + f = f$. □

Lemma 5.15. *If the finite semiring R is $\{0,1\}$-free, then R has a rank-function.*

Proof. For $a, b \in R$ we define $a \preceq b$ if b can be obtained from a by iterated additions and left- and right-multiplications of elements from R. This is equivalent to the following condition:

$$\exists \ell, r, c \in R : b = \ell a r + c \text{ (where each of the elements } \ell, r, c \text{ can be also missing)}$$

Since $\preceq$ is a preorder on R, there is a function $\mathsf{rank} : R \to \mathbb{N} \setminus \{0\}$ such that for all $a, b \in R$ we have

- $\mathsf{rank}(a) = \mathsf{rank}(b)$ iff $a \preceq b$ and $b \preceq a$,
- $\mathsf{rank}(a) \leq \mathsf{rank}(b)$ if $a \preceq b$.

We claim that rank satisfies the conditions of Definition 5.10. The first two conditions are clear, since $a \preceq a + b$ and $a, b \preceq ab$. For the third condition, let $e, f \in E$, $a, b \in eRf$ such that $\mathsf{rank}(a+b) = \mathsf{rank}(a)$, which is equivalent to $a+b \preceq a$. Assume that $a = \ell(a+b)r+c = \ell a r+\ell b r+c$ for some $\ell, r, c \in R$ (the case without c can be handled in the same way). Since $a = eaf$ and $b = ebf$, we have $a = \ell e(a + b)fr + c$ and hence we can assume that ℓ and r are not missing. By multiplying with e from the left and f from the right we get $a = (e\ell e)(a + b)(frf) + (ecf)$, so we can assume that $\ell = e\ell e$ and $r = frf$. After m repeated applications of $a = \ell a r + \ell b r + c$ we obtain

$$a = \ell^m a r^m + \sum_{i=1}^{m} \ell^i b r^i + \sum_{i=0}^{m-1} \ell^i c r^i. \tag{5}$$

Let $n \geq 1$ such that nx is additively idempotent and x^n is multiplicatively idempotent for all $x \in R$. Hence nx^n is both additively and multiplicatively idempotent for all $x \in R$. If we choose $m = n^2$, the right hand side of (5) contains the partial sum $\sum_{i=1}^{n} \ell^{in} b r^{in}$. Furthermore, $e(n\ell^n) = (n\ell^n)e = n\ell^n$ and $f(nr^n) = (nr^n)f = nr^n$. Therefore, Lemma 5.14 implies that $n\ell^n = n\ell^n + e$ and $nr^n = nr^n + f$, and hence:

$$\begin{aligned}
\sum_{i=1}^{n} \ell^{in} b r^{in} &= n(\ell^n b r^n) = n^2(\ell^n b r^n) = (n\ell^n)b(nr^n) = (n\ell^n + e)b(nr^n) \\
&= (n\ell^n)b(nr^n) + eb(nr^n) = (n\ell^n)b(nr^n) + eb(nr^n + f) \\
&= (n\ell^n)b(nr^n) + eb(nr^n) + ebf = \left(\sum_{i=1}^{n} \ell^{in} b r^{in}\right) + b.
\end{aligned}$$

Thus, we can replace in (5) the partial sum $\sum_{i=1}^{n} \ell^{in} b r^{in}$ by $\sum_{i=1}^{n} \ell^{in} b r^{in} + b$, which proves that $a = a + b$. □

6 An application to formal language theory

We present an application of our complexity results for circuit evaluation to formal language theory. Recall that a *context-free grammar* (over Σ) is a tuple $\mathcal{G} = (V, \Sigma, S, P)$ consisting of a finite set of variables V, a finite alphabet Σ, a start variable $S \in V$ and a set P of productions $A \to \alpha$ where $A \in V$ and $\alpha \in (V \cup \Sigma)^*$. We write $L_{\mathcal{G}}(A)$ for the language of A, i.e. the set of words $w \in \Sigma^*$ which can be derived from A using the productions in P, and write $L(\mathcal{G})$ for $L_{\mathcal{G}}(S)$. Every circuit over the free monoid Σ^* can be seen as a context-free grammar producing exactly one word. Such a circuit is also called a *straight-line program*, briefly SLP. It is a context-free grammar $\mathcal{H} = (V, \Sigma, S, P)$ that contains for every variable $A \in V$ exactly one rule of the form $A \to \alpha$. Moreover, $\mathcal{H}$ is acyclic, i.e., there is no non-empty derivation from a variable A to a word containing A. We denote with $\mathsf{val}_{\mathcal{H}}(A)$ the unique word in the language $L_{\mathcal{H}}(A)$. Moreover, let $\mathsf{val}(\mathcal{H}) = \mathsf{val}_{\mathcal{H}}(S)$.

Given an alphabet Σ and a language $L \subseteq \Sigma^*$, the *intersection non-emptiness problem for* L, denoted by $\mathsf{CFG\text{-}IP}(L, \Sigma)$, is the following decision problem:

Input: A context-free grammar $\mathcal{G}$ over Σ
Question: Does $L(\mathcal{G}) \cap L \neq \emptyset$ hold?

For every regular language L, this problem is solvable in polynomial-time by constructing a context-free grammar for $L(\mathcal{G}) \cap L$ from the given grammar $\mathcal{G}$ and a finite automaton for L. The constructed grammar then has to be tested for emptiness, which is possible in polynomial time. However, testing emptiness of a given context-free language is P-complete [16]. An easy reduction shows that the problem $\mathsf{CFG\text{-}IP}(L, \Sigma)$ is P-complete for any non-empty language L.

Theorem 6.1. *For every non-empty language $L \subseteq \Sigma^*$, $\mathsf{CFG\text{-}IP}(L, \Sigma)$ is P-complete.*

Proof. Let $\mathcal{G} = (V, \Sigma, S, P)$ be a context-free grammar. We reduce emptiness of $\mathcal{G}$ to the intersection non-emptiness problem as follows. Let $X \notin V$ be a new variable. We replace all occurrences of terminal symbols in productions of $\mathcal{G}$ by X and then add the rules $X \to \varepsilon$ and $X \to aX$ for all $a \in \Sigma$ (thus, X produces Σ^*). Observe that the new grammar $\mathcal{G}'$ satisfies $L(\mathcal{G}) \neq \emptyset$ if and only if $L(\mathcal{G}') = \emptyset$. Further, $L(\mathcal{G}')$ is either $\emptyset$ or Σ^*. Hence, $L(\mathcal{G}) \neq \emptyset$ if and only if $L(\mathcal{G}') \cap L \neq \emptyset$. Clearly, the reduction can be performed in logspace. □

By Theorem 6.1 we have to put some restriction on context-free grammars in order to get NC-algorithms for the intersection non-emptiness problem. It turns out that productivity of all variables is the right assumption. Thus, we require that $L_{\mathcal{G}}(A) \neq \emptyset$ for all $A \in V$. In order to avoid a promise problem (testing productivity of a variable is P-complete) we add to the input grammar $\mathcal{G} = (V, \Sigma, S, P)$ an SLP $\mathcal{H} = (V, \Sigma, S, R)$ which *uniformizes* $\mathcal{G}$ in the sense that R contains for every variable $A \in V$ exactly one rule $(A \to \alpha) \in P$. Hence, the word $\mathsf{val}_{\mathcal{H}}(A) \in L_{\mathcal{G}}(A)$ is a witness for $L_{\mathcal{G}}(A) \neq \emptyset$.

Example 6.2. *Here is a context-free grammar, where the underlined productions form a uniformizing SLP:*

$$S \to SS,\ S \to aSb,\ \underline{S \to A},\ A \to aA,\ \underline{A \to B},\ B \to bB,\ \underline{B \to b}$$

We study the following decision problem $\mathsf{PCFG\text{-}IP}(L, \Sigma)$ in the rest of this section:

Input: A productive context-free grammar $\mathcal{G}$ over Σ and a uniformizing SLP $\mathcal{H}$ for $\mathcal{G}$.
Question: Does $L(\mathcal{G}) \cap L \neq \emptyset$ hold?

The goal of this section is to classify regular languages L by the complexity of $\mathsf{PCFG\text{-}IP}(L, \Sigma)$.

6.1 Reduction to circuit evaluation

In the following we prove that $\mathsf{PCFG\text{-}IP}(L, \Sigma)$ is equivalent (with respect to constant depth reductions) to the circuit evaluation problem for a suitable finite semiring that is derived from L.

We start with a few standard notations from algebraic language theory. A language $L \subseteq \Sigma^*$ is *recognized* by a monoid M if there exists a homomorphism $h : \Sigma^* \to M$ such that $h^{-1}(F) = L$ for some $F \subseteq M$. It is known that a language is regular if and only if it is recognized by a finite monoid. The *syntactic congruence* $\equiv_L$ is the equivalence relation on Σ^* that is defined by $u \equiv_L v$ $(u, v \in \Sigma^*)$ if the following holds: $\forall x, y \in \Sigma^* : xuy \in L \Leftrightarrow xvy \in L$. It is indeed a congruence relation on the free monoid Σ^*. The quotient monoid $\Sigma^*/\equiv_L$ is the smallest monoid which recognizes L; it is called the *syntactic monoid of* L. From now on we fix a language $L \subseteq \Sigma^*$, a surjective homomorphism $h : \Sigma^* \to M$ onto the syntactic monoid M of L and a set $F \subseteq M$ satisfying $h^{-1}(F) = L$. As a variation of computation problem $\mathsf{CEP}(\mathcal{P}(M))$, we define the decision problem $\mathsf{CEP}(\mathcal{P}(M), F)$:

Input: A circuit over $\mathcal{P}(M)$
Question: Does $[\mathcal{C}] \cap F \neq \emptyset$ hold?

Lemma 6.3. $\mathsf{PCFG\text{-}IP}(L, \Sigma)$ *is equivalent to* $\mathsf{CEP}(\mathcal{P}(M), F)$ *with respect to constant depth reductions.*

Proof. We first reduce $\mathsf{PCFG\text{-}IP}(L, \Sigma)$ to $\mathsf{CEP}(\mathcal{P}(M), F)$. Let $\mathcal{G} = (V, \Sigma, S, P)$ be a productive context-free grammar and let $\mathcal{H} = (V, \Sigma, S, R)$ be a uniformizing SLP for $\mathcal{G}$. To decide whether $L(\mathcal{G}) \cap L \neq \emptyset$, we construct a circuit whose gates compute all sets $X_A = h(L_{\mathcal{G}}(A)) \in \mathcal{P}(M)$ for $A \in V$. Then we test whether X_S intersects F.

In preparation we compute from $\mathcal{H}$ a circuit whose gates evaluate to the singleton sets $X_A^{(0)} := \{h(\mathsf{val}_{\mathcal{H}}(A))\}$ for $A \in V$. Every production $(A \to \alpha_0 A_1 \alpha_1 \cdots A_k \alpha_k) \in R$ with $A_1, \ldots, A_k \in V$ and $\alpha_0, \ldots, \alpha_k \in \Sigma^*$ is translated to the definition

$$\mathsf{rhs}(A) = \{h(\alpha_0)\} \cdot A_1 \cdot \{h(\alpha_1)\} \cdots A_k \cdot \{h(\alpha_k)\}$$

in the circuit. Now the tuple $(X_A)_{A \in V}$ is the least fixed-point of the following monotone operator μ:

$$\mu : \left(2^M\right)^{|V|} \to \left(2^M\right)^{|V|} \tag{6}$$

$$\mu((Y_A)_{A \in V}) = \Big(Y_A \cup \bigcup h(\alpha_0) Y_{A_1} h(\alpha_1) \cdots Y_{A_k} h(\alpha_k)\Big)_{A \in V} \tag{7}$$

where the union in (7) ranges over all productions $A \to \alpha_0 A_1 \alpha_1 \cdots A_k \alpha_k \in P$ for $A_1, \ldots, A_k \in V$ and $\alpha_0, \ldots, \alpha_k \in \Sigma^*$. The smallest fixpoint of μ can be computed by the fixed-point iteration

$$Y_A^{(0)} = \emptyset, \quad Y_A^{(n+1)} = \mu((Y_A^{(n)})_{A \in V}) \tag{8}$$

which reaches the least fixed-point after at most $|V| \cdot |M|$ steps. Equation (8) gives rise to an AC^0-computable circuit over the semiring $(2^M, \cup, \cdot)$ computing $X_S = h(L(\mathcal{G}))$. Since we disallow the empty set in $\mathcal{P}(M)$, we instead initialize the fixed-point iteration in (8) with the non-empty subsets $X_A^{(0)} \subseteq X_A$. This yields a circuit over $\mathcal{P}(M)$ for X_S.

Let us now reduce $\mathsf{CEP}(\mathcal{P}(M), F)$ to $\mathsf{PCFG\text{-}IP}(L, \Sigma)$. Let $\mathcal{C} = (V, A_0, \mathsf{rhs})$ be a circuit over $\mathcal{P}(M)$. We define a grammar $\mathcal{G} = (V, \Sigma, A_0, P)$ as follows:

- If $\mathsf{rhs}(A) = \{m_1, \ldots, m_k\} \in \mathcal{P}(M)$, add the rules $A \to w_i$ to P $(1 \leq i \leq k)$ where $w_i \in \Sigma^*$ is any word with $h(w_i) = m_i$.
- If $\mathsf{rhs}(A) = B \cup C$, add the rules $A \to B$ and $A \to C$ to P.
- If $\mathsf{rhs}(A) = B \cdot C$, add the rules $A \to BC$ to P.

Then every gate $A \in V$ evaluates to $h(L_{\mathcal{G}}(A))$. In particular, we have $h(L(\mathcal{G})) = [\mathcal{C}]$. Therefore, $[\mathcal{C}] \cap F \neq \emptyset$ if and only if $L(\mathcal{G}) \cap L \neq \emptyset$. □

Now clearly $\mathsf{CEP}(\mathcal{P}(M), F)$ is logspace reducible to $\mathsf{CEP}(\mathcal{P}(M))$ but not necessarily vice versa as the following example shows:

Example 6.4. *Consider the language $L = \{a,b\}^* a \{a,b\}^* \subseteq \{a,b\}^*$ of all words which contain the symbol a. Its syntactic monoid is the two-element monoid $M = \{1, e\}$ where e is an idempotent element. We have $L = h^{-1}(\{e\})$ for the homomorphism $h : \{a,b\}^* \to M$ defined by $h(a) = e$, $h(b) = 1$. One can decide $\mathsf{CEP}(\mathcal{P}(M), \{e\})$ in NL: For a circuit $\mathcal{C}$ we have $e \in [\mathcal{C}]$ if and only if an input gate with e on the right-hand side is reachable from the output gate. However, since M is not a local group, $\mathsf{CEP}(\mathcal{P}(M))$ is P-complete by Theorem 4.8. This can be also seen directly: The sets $\{e\}$ and $\{1, e\}$ form a Boolean semiring. On the other hand, for the purpose of deciding $\mathsf{CEP}(\mathcal{P}(M), \{e\})$ one does not have to distinguish the sets $\{e\}$ and $\{1, e\}$. Identifying these two sets in $\mathcal{P}(M)$ yields a $\{0,1\}$-free semiring whose circuit evaluation problem is in NL.*

The example above motivates to define a congruence relation on $\mathcal{P}(M)$ where congruent subsets are either both disjoint from F or both not. Define the equivalence relation $\sim_F$ on $\mathcal{P}(M)$ by

$$A_1 \sim_F A_2 \iff \forall \ell, r \in M \colon \ell A_1 r \cap F \neq \emptyset \iff \ell A_2 r \cap F \neq \emptyset$$

for subsets $A_1, A_2 \in \mathcal{P}(M)$. The following lemmata summarize the basic properties of $\sim_F$.

Lemma 6.5. *The following properties hold.*

(1) $A_1 \sim_F A_2$ implies $(LA_1R \cap F \neq \emptyset \iff LA_2R \cap F \neq \emptyset)$ for all $L, R \subseteq M$.

(2) The relation $\sim_F$ is a congruence relation. In particular, $\mathcal{P}(M)/\!\sim_F$ is a semiring.

(3) Every $\sim_F$-class contains a largest subset with respect to $\subseteq$.

Proof. Property (1) is clear because $LA_iR \cap F \neq \emptyset$ if and only if $\ell A_i r \cap F \neq \emptyset$ for some $\ell \in L, r \in R$. For (2), assume $A_1 \sim_F A_2$ and $B_1 \sim_F B_2$. Then for all $\ell, r \in M$ we have

$$\begin{aligned}
\ell(A_1 \cup B_1)r \cap F \neq \emptyset &\iff (\ell A_1 r \cup \ell B_1 r) \cap F \neq \emptyset \\
&\iff \ell A_1 r \cap F \neq \emptyset \quad \text{or} \quad \ell B_1 r \cap F \neq \emptyset \\
&\iff \ell A_2 r \cap F \neq \emptyset \quad \text{or} \quad \ell B_2 r \cap F \neq \emptyset \\
&\iff (\ell A_2 r \cup \ell B_2 r) \cap F \neq \emptyset \\
&\iff \ell(A_2 \cup B_2)r \cap F \neq \emptyset
\end{aligned}$$

and, by (1),

$$\ell A_1 (B_1 r) \cap F \neq \emptyset \iff (\ell A_2) B_1 r \cap F \neq \emptyset \iff \ell A_2 B_2 r \cap F \neq \emptyset.$$

For (3) note that by (2)l, $A_1 \sim_F A_2$ implies $A_1 = A_1 \cup A_1 \sim_F A_1 \cup A_2$. Thus, every $\sim_F$-class is closed under union and therefore has a largest element with respect to $\subseteq$. □

Lemma 6.6. $\mathsf{CEP}(\mathcal{P}(M), F)$ *is equivalent to* $\mathsf{CEP}(\mathcal{P}(M)/\!\sim_F)$ *with respect to constant depth Turing-reductions. In other words:* $\mathsf{AC}^0(\mathsf{CEP}(\mathcal{P}(M), F)) = \mathsf{AC}^0(\mathsf{CEP}(\mathcal{P}(M)/\!\sim_F))$.

Proof. Clearly, every circuit $\mathcal{C}$ over $\mathcal{P}(M)$ can be regarded as a circuit $\mathcal{C}'$ over $\mathcal{P}(M)/\!\sim_F$ such that $[\mathcal{C}']$ is the $\sim_F$-class of $[\mathcal{C}]$. Every $\sim_F$-class either contains only subsets of M which are disjoint to F or only subsets with non-empty intersection with F. Thus, $[\mathcal{C}']$ determines whether $[\mathcal{C}] \cap F \neq \emptyset$.

For the other direction, given a circuit $\mathcal{C}'$ over $\mathcal{P}(M)/\!\sim_F$, we define a circuit $\mathcal{C}$ over $\mathcal{P}(M)$ by picking arbitrary representative elements (subsets of M) for the input values (which are $\sim_F$-classes) of the circuit $\mathcal{C}'$. Then we test for all $\ell, r \in M$ whether $\ell[\mathcal{C}]r \cap F \neq \emptyset$. This information is independent from the choice of representative elements and uniquely determines the $\sim_F$-class $[\mathcal{C}']$. □

From Corollary 4.6 and Lemma 6.3 and 6.6 we obtain:

Theorem 6.7. $\mathsf{PCFG\text{-}IP}(L, \Sigma)$ *is equivalent to* $\mathsf{CEP}(\mathcal{P}(M)/\!\sim_F)$ *with respect to constant depth Turing-reductions. Therefore,*

- PCFG-IP(L, Σ) *is* P*-complete if* $\mathcal{P}(M)/\sim_F$ *is not* $\{0,1\}$*-free or its multiplicative semigroup is not solvable,*
- PCFG-IP(L, Σ) *is in* DET *if* $\mathcal{P}(M)/\sim_F$ *is* $\{0,1\}$*-free and its multiplicative semigroup is solvable, and*
- PCFG-IP(L, Σ) *is in* NL *if* $\mathcal{P}(M)/\sim_F$ *is* $\{0,1\}$*-free and its multiplicative semigroup is aperiodic.*

It would be nice to have a simple characterization of when $\mathcal{P}(M)/\sim_F$ is $\{0,1\}$-free (resp., its multiplicative semigroup is solvable). For $\{0,1\}$-freeness, we can show:

Proposition 6.8. *$\mathcal{P}(M)/\sim_F$ is $\{0,1\}$-free if and only if*

$$\forall s,t \in M,\, e \in E(M) : st \in F \implies set \in F. \tag{9}$$

Proof. Assume first that $\mathcal{P}(M)/\sim_F$ is $\{0,1\}$-free. Let $s,t \in M$ such that $st \in F$ and let $e \in E(M)$. We have $\{e\} \sim_F \{1,e\}$ since otherwise their two $\sim_F$-classes would form a Boolean subsemiring $\mathbb{B}_2$ in $\mathcal{P}(M)/\sim_F$. From $st \in F$ it follows that $s\{1,e\}t \cap F \neq \emptyset$ and hence $s\{e\}t \cap F \neq \emptyset$. Therefore $set \in F$.

Assume now that the implication (9) holds and towards a contradiction assume that R is a subsemiring of $\mathcal{P}(M)$ with the zero-element $[A]_{\sim_F}$ and the one-element $[B]_{\sim_F}$. We choose A, B to be the $\subseteq$-maximal elements in their classes. Then we have $A \subseteq B$ because $[A]_{\sim_F} \cup [B]_{\sim_F} = [B]_{\sim_F}$. Further $A^2 \sim_F A$ implies $A^2 \subseteq A$ by maximality of A and therefore A contains at least an idempotent $e \in A$ (take a^ω for any $a \in A$). Finally we have $AB \sim_F A$, which implies $AB \subseteq A$.

Since A and B are not $\sim_F$-equivalent there exist elements $s,t \in M$ such that $sAt \cap F = \emptyset$ but $sbt \in F$ for some $b \in B$. However, $eb \in AB \subseteq A$ and by assumption $sebt \in F$, contradiction. □

We do not have a nice characterization for solvability of the multiplicative semigroup of $\mathcal{P}(M)/\sim_F$.

Let us conclude this section with an application of Corollary 6.7:

Example 6.9. *Consider a language of the form $L = \Sigma^* a_1 \Sigma^* a_2 \Sigma^* \dots a_k \Sigma^*$ for $a_1, \dots, a_k \in \Sigma$, which is a so called* piecewise testable language*. We claim that* PCFG-IP(L, Σ) *is decidable in* NL*. First, since $uw \in L$ implies $uvw \in L$ for all $u, v, w \in \Sigma^*$, the syntactic monoid M and the accepting subset $F \subseteq M$ of L clearly satisfies the condition of Proposition 6.8. Second, clearly $\mathcal{P}(M)_+$ is aperiodic and hence also $(\mathcal{P}(M)/\sim_F)_+$. Third, we show that $\mathcal{P}(M)_\bullet$ and hence also $(\mathcal{P}(M)/\sim_F)_\bullet$ is aperiodic. Simon's theorem [32] states that a language is piecewise testable if and only if its syntactic monoid is $\mathcal{J}$-trivial. We claim that $\mathcal{P}(M)_\bullet$ is also $\mathcal{J}$-trivial, in particular aperiodic. Let $A, B \in \mathcal{P}(M)$ such that $A \equiv_{\mathcal{J}} B$. Consider the directed bipartite graph on $A \uplus B$ with edges*

$$\{(a,b) \in A \times B \mid a \geq_{\mathcal{J}} b\} \cup \{(b,a) \in B \times A \mid b \geq_{\mathcal{J}} a\}.$$

Every vertex has at least one outgoing and one incoming edge, which means that it belongs to a non-trivial strongly connected component. Since M is $\mathcal{J}$-trivial, we must have $A = B$.

7 Some results about infinite semirings

It would be interesting to see, whether our techniques can be extended to certain infinite semirings. Recently, it was shown that for certain finitely generated (but infinite) linear groups, the circuit evaluation problem belongs to NC^2 or at least coRNC^2 (the complement of the randomized version of NC^2) [17, 18]. Of course, if one deals with infinite structures, one needs a finite representation of the elements of the structure. Moreover, in contrast to finite structures, the circuit evaluation problem (i.e., the question, whether a given circuit evaluates to a given element) is not equivalent to its computation variant, where one wants to compute the output value of the circuit. A good example is the arithmetic ring $(\mathbb{Z}, +, \cdot)$. Whether a given circuit over this ring evaluates to a

given element is equivalent to the question, whether a given circuit evaluates to zero, which, in turn, is equivalent to the famous polynomial identity testing problem [2]. Its complexity is in co-randomized polynomial time and no deterministic polynomial time algorithm is known (the problem is easily seen to be P-hard). On the other hand, by iterated squaring one can construct a circuit over $(\mathbb{Z}, +, \cdot)$ with n gates that evaluates to 2^{2^n}. The binary representation of this number needs 2^n bits. This shows that in general, we cannot even write down the output value of a given circuit in polynomial time. But even for semirings, where this phenomenon does not occur, it seems to be difficult to obtain NC-algorithms. As an example, let us consider one of the simplest infinite semirings, namely the max-plus semiring $(\mathbb{N}, \max, +)$. Note that it is $\{0,1\}$-free (this follows from Lemma 3.1) and $+$ (the semiring multiplication) is commutative.

Theorem 7.1. $\mathsf{CEP}(\mathbb{N}, \max, +)$ *is* P*-complete.*

Proof. A given circuit over $(\mathbb{N}, \max, +)$ can be sequentially evaluated in polynomial time by representing all numbers in binary notation. Note that a max-gate does not increase the number of bits, whereas a $+$-gate can increase the number of bits by at most one (the sum of an n-bit number and an m-bit number has at most $\max\{n, m\} + 1$ many bits).

For the lower bound we reduce from $\mathsf{CEP}(\mathbb{B}_2)$. Let $\mathcal{C} = (V, A_0, \mathsf{rhs}_\mathcal{C})$ be a circuit over the boolean semiring. W.l.o.g. we can assume that $\mathcal{C}$ consists of n layers, where all wires go from layer k to layer $k+1$ for some k. Layer 1 contains the input gates and layer n contains the output gate A_0. We now construct a circuit $\mathcal{D} = (V, A_0, \mathsf{rhs}_\mathcal{D})$ over $(\mathbb{N}, \max, +)$ with the same gates as $\mathcal{C}$. The idea is to construct $\mathcal{D}$ such that the following conditions hold for every gate $A \in V$ on layer k:

(a) If $[A]_\mathcal{C} = 0$ then $[A]_\mathcal{D} = 2^k - 1$

(b) If $[A]_\mathcal{C} = 1$ then $[A]_\mathcal{D} = 2^k$

To get this correspondence, we define the right-hand sides for $\mathcal{D}$ as follows, where B and C are on layer $k < n$ and A is on layer $k+1$.

- If $\mathsf{rhs}_\mathcal{C}(A) = B \wedge C$ then $\mathsf{rhs}_\mathcal{D}(A) = \max(B + C, 2^{k+1} - 1)$.
- If $\mathsf{rhs}_\mathcal{C}(A) = B \vee C$ then $\mathsf{rhs}_\mathcal{D}(A) = \max(B, D) + 2^k$.

Moreover, if $\mathsf{rhs}_\mathcal{C}(A) = 0$ (resp., $\mathsf{rhs}_\mathcal{C}(A) = 1$), then gate A is on layer 1 and we set $\mathsf{rhs}_\mathcal{D}(A) = 1$ (resp., $\mathsf{rhs}_\mathcal{D}(A) = 2$). With these settings, it is straightforward to show that (a) and (b) hold. In particular, we have $[\mathcal{C}] = 1$ if and only if $[\mathcal{D}] = 2^n$. □

Let us conclude this section with a few results on power semirings, in particular, power semirings of groups. Recall from Example 4.7 that for a finite solvable group G, the semiring of non-empty subsets $\mathcal{P}(G)$ has a circuit evaluation problem in DET. This motivates the question for the complexity of the circuit evaluation problem of $\mathcal{P}(G)$ for an infinite but finitely generated group G. To avoid the problem of representing the output value of a circuit over $\mathcal{P}(G)$ (which may be a set of exponentially many group elements) we consider again the variant $\mathsf{CEP}(\mathcal{P}(G), F)$ from Section 6, which asks whether the set computed by a given circuit over $\mathcal{P}(G)$ contains an element from a given set $F \subseteq G$. We can assume that the input gates of the circuit are labelled with singleton sets $\{a\}$, where a is a generator of G. This variant of the circuit evaluation problem can be seen as a nondeterministic version of the compressed word problem for G (i.e., $\mathsf{CEP}(G)$) [21]. As an example, a result from [33] can be reformulated as follows: $\mathsf{CEP}(\mathcal{P}(\mathbb{Z}), \{0\})$ is NP-complete, where $\mathbb{Z}$ is the additive group of integers, i.e., the free group of rank 1. For the free group of rank 2, briefly F_2, we have:

Theorem 7.2. $\mathsf{CEP}(\mathcal{P}(F_2), \{1\})$ *is* PSPACE*-complete.*

Proof. Let us fix the generating set $\{a, a^{-1}, b, b^{-1}\}$ for F_2. We first show that $\mathsf{CEP}(\mathcal{P}(F_2), \{1\})$ is in PSPACE. First, we show that $\mathsf{CEP}(\mathcal{P}(F_2), \{1\})$ restricted to circuits that are trees is in LOGCFL, i.e., it can be solved in polynomial time on a nondeterministic pushdown machine with an auxiliary working tape of logarithmic length. For this, the machine traverses the input circuit (a tree) in

a depth-first left-to-right manner and thereby stores the current node in the tree. Each time, the machine arrives at a $\cup$-gate A from A's parent gate, it nondeterministically decides whether it proceeds to the left or right child of A. If it goes to the right child, then the subtree rooted in the left child of A is omitted in the traversal. Similarly, if the machine goes to the left child, then when returning to A it omits the right subtree of A in the traversal. Thus, the machine chooses nondeterministically for each union gate one of the two children and only traverses the subtree of the chosen child. Finally, when the machine arrives at a leaf of the tree, i.e., an input gate of the circuit, and this leaf is labelled with the group generator $x \in \{a, a^{-1}, b, b^{-1}\}$ then it pushes x on the pushdown. If the top of the pushdown then ends with $x^{-1}x$, it pops $x^{-1}x$ from the pushdown. Then the machine continues with the traversal of the tree. At the end of the traversal, the machine accepts if and only if the pushdown is empty.

Basically, the above machine nondeterministically chooses a word over $\{a, a^{-1}, b, b^{-1}\}$ that belongs to the circuit output if the circuit is interpreted over the power semiring $\mathcal{P}(\{a, a^{-1}, b, b^{-1}\}^*)$. Moreover, while generating this word, it verifies, using the pushdown, that the word is equal to 1 in the free group F_2. Now we use the fact that LOGCFL is contained in $\mathsf{DSPACE}(\log^2(n))$. Hence, $\mathsf{CEP}(\mathcal{P}(F_2), \{1\})$ restricted to circuits that are trees is in $\mathsf{DSPACE}(\log^2(n))$. From this, it follows easily that $\mathsf{CEP}(\mathcal{P}(F_2), \{1\})$ for general (non-tree-like) circuits belongs to PSPACE: The function that maps a circuit $\mathcal{C}$ to its unfolding (a tree that is equivalent to $\mathcal{C}$) can be computed by a Turing machine with output in polynomial space (a so called PSPACE-transducer). This follows from the fact that the gates of the unfolding of $\mathcal{C}$ can be identified with paths in $\mathcal{C}$ that start in the output gate and go down in the circuit. The set of all these paths can be produced in polynomial space. Now we use a simple lemma (see [22, Lemma 1] stating that a preimage $f^{-1}(L)$ belongs to PSPACE, if f can be computed by a PSPACE-transducer and the language L can be decided in polylogarithmic space.

For the lower bound we use a result from [28], saying that the intersection non-emptiness problem for given *acyclic* context-free grammars $\mathcal{G}_1$ and $\mathcal{G}_2$ is PSPACE-complete. By coding terminal symbols into a binary alphabet, we can assume that the terminal alphabet of $\mathcal{G}_1$ and $\mathcal{G}_2$ is $\{a, b\}$. Moreover, by the construction in [28], $\mathcal{G}_1$ and $\mathcal{G}_2$ are productive (which for acyclic grammars just means that every nonterminal has a rule). For a word $w = a_1 a_2 \cdots a_n$ with $a_i \in \{a, b\}$ let $w^{-1} = a_n^{-1} \cdots a_2^{-1} a_1^{-1}$. It is straightforward to construct from $\mathcal{G}_1$ and $\mathcal{G}_2$ an acyclic productive context-free grammar $\mathcal{G}$ over the terminal alphabet $\{a, b, a^{-1}, b^{-1}\}$ such that

$$L(\mathcal{G}) = \{w_1 w_2^{-1} \mid w_1 \in L(\mathcal{G}_1), w_2 \in L(\mathcal{G}_2)\}.$$

Now an acyclic productive context-free grammar with terminal alphabet Σ can be seen as a circuit over the semiring $\mathcal{P}(\Sigma^*)$. By interpreting the circuit for the grammar $\mathcal{G}$ as a circuit $\mathcal{C}$ over $\mathcal{P}(F_2)$ we see that $L(\mathcal{G}_1) \cap L(\mathcal{G}_2) \neq \emptyset$ if and only if $[\mathcal{C}]$ contains 1. □

8 Conclusion and outlook

We proved a dichotomy result for the circuit evaluation problem for finite semirings: If (i) the semiring has no subsemiring with an additive and multiplicative identity and both are different and (ii) the multiplicative subsemigroup is solvable, then the circuit evaluation problem is in $\mathsf{DET} \subseteq \mathsf{NC}^2$, otherwise it is P-complete.

The ultimate goal would be to obtain such a dichotomy for all finite algebraic structures. One might ask whether for every finite algebraic structure $\mathcal{A}$, $\mathsf{CEP}(\mathcal{A})$ is P-complete or in NC. It is known that under the assumption $\mathsf{P} \neq \mathsf{NC}$ there exist problems in $\mathsf{P} \setminus \mathsf{NC}$ that are not P-complete [36]. In [9] it is shown that every circuit evaluation problem $\mathsf{CEP}(\mathcal{A})$ is equivalent to a circuit evaluation problem $\mathsf{CEP}(A, \circ)$, where $\circ$ is a binary operation.

Acknowledgment. We thank Benjamin Steinberg for helpful discussions on semigroups. We are grateful to Volker Diekert for pointing out to us the proof of the implication $(3 \Rightarrow 4)$ in the proof of Lemma 3.1.

References

[1] Manindra Agrawal and Somenath Biswas. Primality and identity testing via chinese remaindering. *Journal of the Association for Computing Machinery*, 50(4):429–443, 2003.

[2] Eric Allender, Peter Bürgisser, Johan Kjeldgaard-Pedersen, and Peter Bro Miltersen. On the complexity of numerical analysis. *SIAM Journal on Computing*, 38(5):1987–2006, 2009.

[3] Eric Allender, Jia Jiao, Meena Mahajan, and V. Vinay. Non-commutative arithmetic circuits: Depth reduction and size lower bounds. *Theor. Comput. Sci.*, 209(1-2):47–86, 1998.

[4] Jorge Almeida, Stuart Margolis, Benjamin Steinberg, and Mikhail Volkov. Representation theory of finite semigroups, semigroup radicals and formal language theory. *Transactions of the American Mathematical Society*, 361(3):1429–1461, 2009.

[5] Carme Àlvarez, José L. Balcázar, and Birgit Jenner. Functional oracle queries as a measure of parallel time. In *Proceedings of the 8th Annual Symposium on Theoretical Aspects of Computer Science, STACS 1991*, volume 480 of *Lecture Notes in Computer Science*, pages 422–433. Springer, 1991.

[6] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[7] Karl Auinger and Benjamin Steinberg. Constructing divisions into power groups. *Theoretical Computer Science*, 341(1–3):1–21, 2005.

[8] Martin Beaudry and Markus Holzer. The complexity of tensor circuit evaluation. *Computational Complexity*, 16(1):60–111, 2007.

[9] Martin Beaudry and Pierre McKenzie. Circuits, matrices, and nonassociative computation. *Journal of Computer and System Sciences*, 50(3):441–455, 1995.

[10] Martin Beaudry, Pierre McKenzie, Pierre Péladeau, and Denis Thérien. Finite monoids: From word to circuit evaluation. *SIAM Journal on Computing*, 26(1):138–152, 1997.

[11] Stephan A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64:2–22, 1985.

[12] Stephen A. Cook and Lila Fontes. Formal theories for linear algebra. *Logical Methods in Computer Science*, 8(1), 2012.

[13] Jonathan S. Golan. *Semirings and their Applications*. Springer, 1999.

[14] Raymond Greenlaw, H. James Hoover, and Walter L. Ruzzo. *Limits to Parallel Computation: P-Completeness Theory*. Oxford University Press, 1995.

[15] Oscar H. Ibarra and Shlomo Moran. Probabilistic algorithms for deciding equivalence of straight-line programs. *Journal of the ACM*, 30(1):217–228, 1983.

[16] Neil D. Jones and William T. Laaser. Complete problems for deterministic polynomial time. *Theor. Comput. Sci.*, 3(1):105–117, 1976.

[17] Daniel König and Markus Lohrey. Evaluating matrix circuits. In *Proceedings of the 21st International Conference on Computing and Combinatorics, COCOON 2015*, volume 9198 of *Lecture Notes in Computer Science*, pages 235–248. Springer, 2015.

[18] Daniel König and Markus Lohrey. Parallel identity testing for skew circuits with big powers and applications. In *Proceedings of the 40th International Symposium on Mathematical Foundations of Computer Science 2015, MFCS 2015, Part II*, volume 9235 of *Lecture Notes in Computer Science*, pages 445–458. Springer, 2015.

[19] S. Rao Kosaraju. On parallel evaluation of classes of circuits. In *Proceedings of the 10th Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 1990*, volume 472 of *Lecture Notes in Computer Science*, pages 232–237. Springer, 1990.

[20] Richard E. Ladner. The circuit value problem is log space complete for P. *SIGACT News*, 7(1):18–20, 1975.

[21] Markus Lohrey. *The Compressed Word Problem for Groups*. SpringerBriefs in Mathematics. Springer, 2014.

[22] Markus Lohrey and Christian Mathissen. Isomorphism of regular trees and words. *Information and Computation*, 224:71–105, 2013.

[23] Donald J. McCarthy and David L. Hayes. Subgroups of the power semigroup of a group. *Journal of Combinatorial Theory, Series A*, 14(2):173–186, 1973.

[24] Pierre McKenzie and Klaus W. Wagner. The complexity of membership problems for circuits over sets of natural numbers. *Computational Complexity*, 16(3):211–244, 2007.

[25] Gary L. Miller, Vijaya Ramachandran, and Erich Kaltofen. Efficient parallel evaluation of straight-line code and arithmetic circuits. *SIAM J. Comput.*, 17(4):687–695, 1988.

[26] Gary L. Miller and Shang-Hua Teng. The dynamic parallel complexity of computational circuits. *SIAM J. Comput.*, 28(5):1664–1688, 1999.

[27] Cristopher Moore, Denis Thérien, François Lemieux, Joshua Berman, and Arthur Drisko. Circuits and expressions with nonassociative gates. *J. Comput. Syst. Sci.*, 60(2):368–394, 2000.

[28] Mark-Jan Nederhof and Giorgio Satta. The language intersection problem for non-recursive context-free grammars. *Information and Computation*, 192(2):172–184, 2004.

[29] John Rhodes and Benjamin Steinberg. *The q-theory of Finite Semigroups*. Springer, 2008.

[30] Alexander A. Rubtsov and Mikhail N. Vyalyi. Regular realizability problems and context-free languages. In *Proceedings of the 17th International Workshop on Descriptional Complexity of Formal Systems, DCFS 2015*, volume 9118 of *Lecture Notes in Computer Science*, pages 256–267. Springer, 2015.

[31] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010.

[32] Imre Simon. Piecewise testable events. In *Proceedings of the 2nd GI Conference on Automata Theory and Formal Languages, 1975*, volume 33 of *Lecture Notes in Computer Science*, pages 214–222. Springer, 1975.

[33] Larry J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the 5th Annual ACM Symposium on Theory of Computing (STOCS 73)*, pages 1–9. ACM Press, 1973.

[34] Stephen D. Travers. The complexity of membership problems for circuits over sets of integers. *Theor. Comput. Sci.*, 369(1-3):211–229, 2006.

[35] Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12(4):641–644, 1983.

[36] Heribert Vollmer. The gap-language-technique revisited. In *Proceedings of the 4th Workshop on Computer Science Logic, CSL '90*, volume 533 of *Lecture Notes in Computer Science*, pages 389–399. Springer, 1990.

[37] Heribert Vollmer. *Introduction to Circuit Complexity*. Springer, 1999.